W90221X

32-bit Processor for Embedded Applications

Version 0.6
January, 2001

The information in this document has not been carefully checked and is believed to be errata prevailing as of the date of publication. Winbond Electronics Corp. reserves the right to make changes in the product or specification, or both, presented in this publication at any time without prior notice. Winbond Electronics Corp. will continue updating this document and make efforts to maintain its contents as accurate as possible. Winbond assumes no responsibility or liability arising from the specification listed herein. Winbond makes no representations that the use of its products in the manner described in this publication will not infringe on existing or future patents, trademark, copyright, or rights of third parties. No license is granted by implication, other under any patent, or patent rights of Winbond Electronics Corp. All other trademarks and registered trademarks are the property of their respective holders.

Revision Summary

Version 0.5

- Architecture AIO was renamed to WIO since this edition.
- Pin designations XGLBCS#, AIOCS#, XROMCS#, XWR#, and XRD were changed to WGLBCS#, WIOCS#, WROMCS#, WWR#, and WRD#, respectively.
- The support of EDO DRAM memory type was de-emphasized since this edition.

Version 0.6

- Pin designation PWRONC was changed to RESET.
- No EDO DRAM and 64-bit DIMM support.
- Built-in TV encoder can work with PAL-M.

Table of Contents

Table of Figures

1 GENERAL DESCRIPTION

The W90221X is a highly integrated 32-bit processor for a wide range of embedded applications, such as set-top box, web browser and visual/data communication devices. Fig 1-1 shows a block diagram of the overall system. The W90221X consists of the system support logics as well as an embedded 32-bit PA-RISC processor.

The 32-bit PA-RISC core has 4K bytes of instruction cache memory, 4K bytes of data cache memory, a dual-cycle multiply/accumulate module, and integrated functions for interfacing to numerous system components and external I/O modules. Besides, it's designed with a flexible power management scheme (under software control) and lots of low power circuits to eliminate the chip's power consumption

The 2-D graphic accelerator is the major mega-functional cell integrated in this chip. This unit provides directly connect to TV, analog LCD monitor and CRT monitors, intending for low cost web browser solution. The chip incorporates an ISA-like bus interface (shared with PCI bus pins) to connect to low speed devices, such as code/data ROM/Flash and traditional ISA-like or IDE devices. The device also includes a PC100 compliant SDRAM controller, a PCI bridge supporting up to four PCI masters, a parallel port interface (PPI), two RS-232 type universal asynchronous serial ports (UART), two timer channels and a flexible synchronous interface (SSI) connecting to an external audio or telephony CODEC devices. The overall features are listed in section 2.

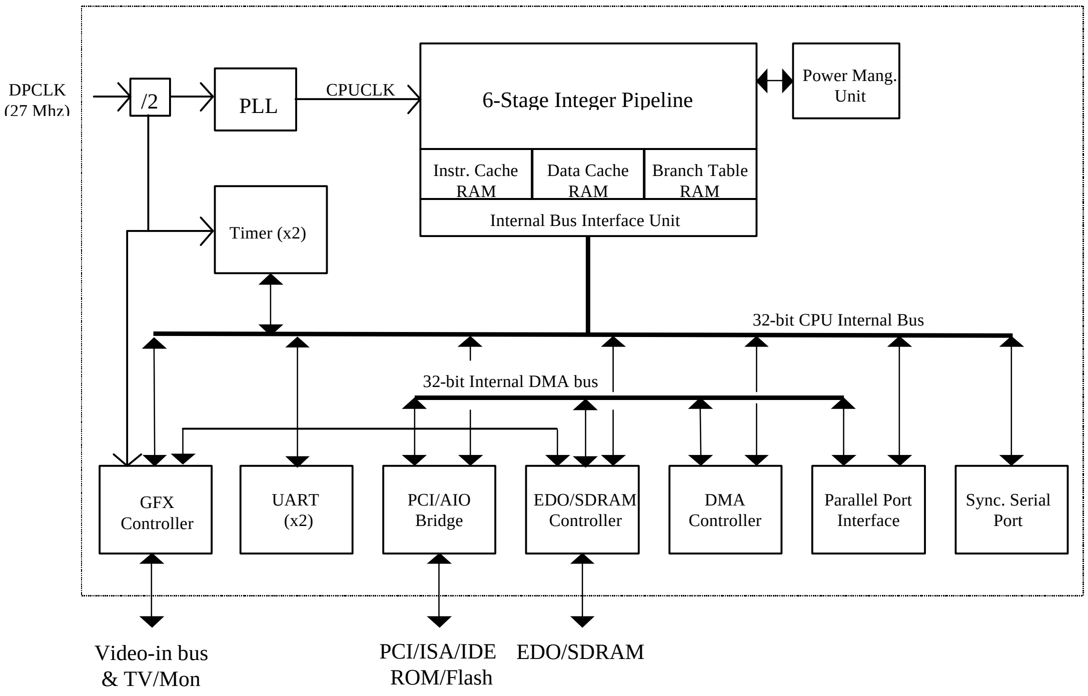


Figure 1-1 W90221X Internal Block Diagram

2 FEATURES

□ HQFP 208-pin package

□ High level of integration

- Minimum number of inter-chip connections
- One 32-bit PA-RISC core with cache memory and multiply-accumulate module
- One 2-D display controller that can directly connect to TV, LCD or monitor
- One ISA-like bus interface to connect 8-bit ROM/Flash, and 8/16-bit ISA device
- One memory controller that supports PC-100 SDRAM
- One PCI bridge that supports up to four PCI master devices
- One parallel port that can be run in the ECP mode as defined in the IEEE 1284 standard
- One RS-232 compliant serial ports to connect external MODEM or other device
- One debugging console port
- One synchronous serial port connecting external audio or telephony Coder/Decoder devices
- Two timer channels for general purpose usage

□ High performance and low power consumption

- 0.35-micron single-poly-triple-metal CMOS process
- Pure 3.3V logics within SDRAM interface
- Split rail design (3.3V/5V IO and 3.3V core) in other interfaces
- Operating frequency up to 100 MHz
- Fully static design
- Real time clock and UART baud rate based on 13.5 MHz or 18.432 MHz

3 PIN CONFIGURATION

The W90221X is packaged in a 208-pin HQFP. The pin configuration is as shown in Figure 3.1.

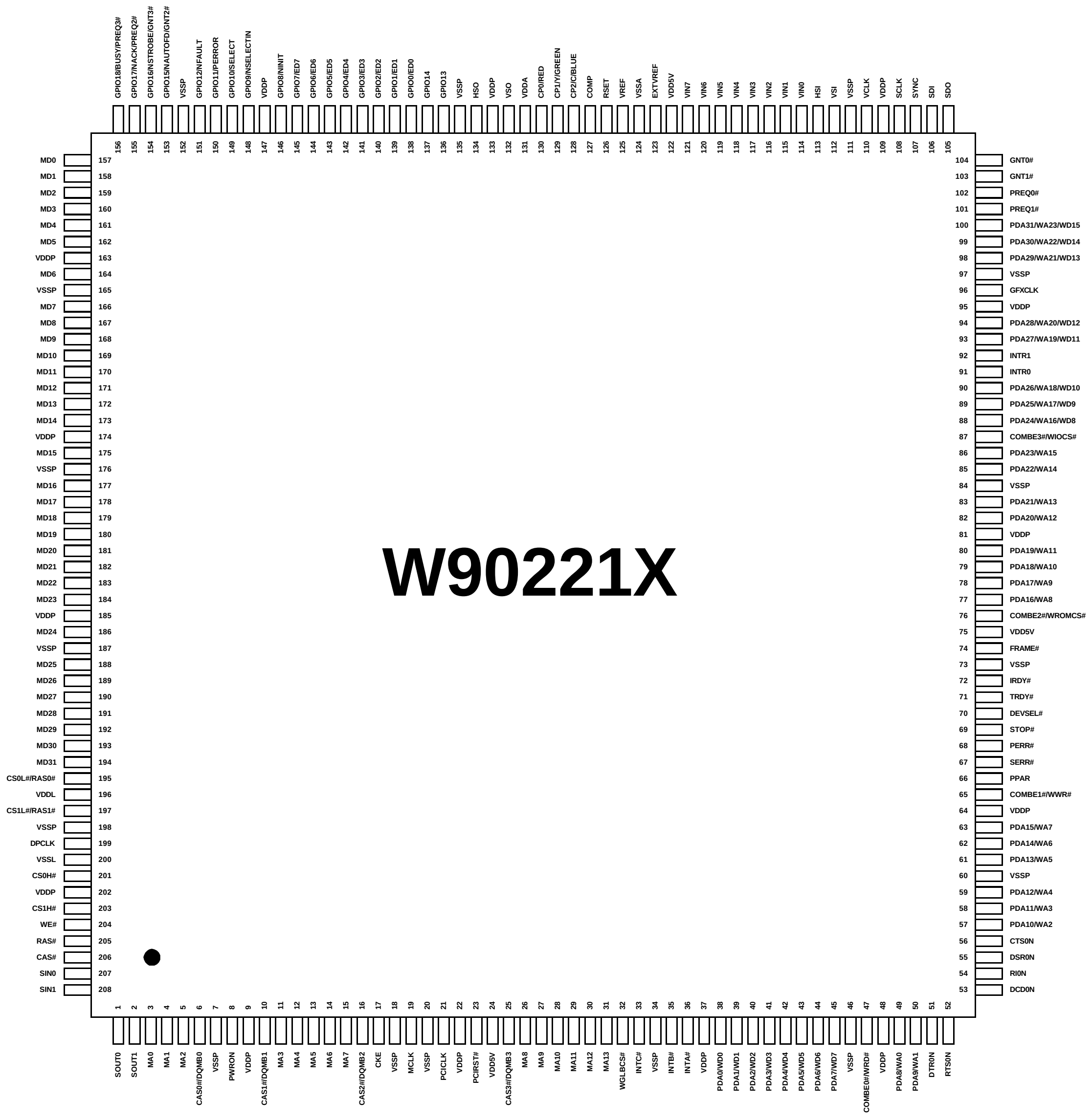


Figure 3.1 W90221X Pin Configuration

4 Pin Description

The following tables provide a brief description of each pin on the W9961CF. The following signal type definitions are used in these descriptions:

I	Input pin
I/O	Bi-directional input/output pin
O	Output pin
TS	Tri-State output pin
OC	open-collector pin
AO	analog output pin
AI	analog input pin

PIN Name	TYPE	PIN #	DESCRIPTION
System Reset and Clock			
RESET	I	8	CPU Power-On reset input, high active
DPCLK	I	199	This clock source serves as internal PLL input as well as VA's system clock. A precise **27MHz** clock source shall be connected to this pin during normal operation.
GFXCLK	I	96	This clock source serves as pixel clock, 36MHz to 50MHz, using in 800x600 non-interlace monitor. For TV subsystem, this clock may pull high or low externally. Meanwhile, GFXCLK may also serves as system OSC (for baud rate or timer adjustment), if **MD[24]** is pull high externally.
General Purpose I/O			
GPIO[0:7]	I/O	138-145	If parallel port is enabled (port 0x3e[4] = 1), these pins serve as bi-directional ECP data bus "**ED[0:7]** " with ED[0] the most significant bit (In/Out). If parallel port is not enabled (port 0x3e[4:5] = 0x), these pins provide general purpose I/O function (In/Out).
GPIO[8]	I/O	146	If parallel port is enabled (port 0x3e[4:5] = 1x), this pin serves as ECP "**nInit**" (output). If parallel port is not enabled (port 0x3e[4:5] = 0x), this pin provides general purpose I/O function (In/Out)
GPIO[9]	I/O	148	If parallel port is enabled (port 0x3e[4:5] = 1x), this pin serves as ECP “**nSelectIn**" (output). If parallel port is not enabled (port 0x3e[4:5] = 0x), this pin provides general purpose I/O function (In/Out)
GPIO[10]	I/O	149	If parallel port is enabled (port 0x3e[4] = 1x), this pin serves as ECP “**Select**" (input). If parallel port is not enabled (port 0x3e[4:5] = 00), this pin provides general purpose I/O function (In/Out). During "clock test" mode (port 0x3e[4] = 01), this pin outputs internal **CPUCLK** (output).
GPIO[11]	I/O	150	If parallel port is enabled (port 0x3e[4] = 1x), this pin serves as ECP "**PError**" (input). If parallel port is not enabled (port 0x3e[4:5] = 00), this pin provides general purpose I/O function (In/Out). During "clock test" mode (port 0x3e[4] = 01), this pin outputs internal **MCLK_CTL** (output).

GPIO[12]	I/O	151	If parallel port is enabled (port 0x3e[4] = 1x), this pin serves as ECP "**nFault**" (input). If parallel port is not enabled (port 0x3e[4:5] = 00), this pin provides general purpose I/O function (In/Out). During "clock test" mode (port 0x3e[4] = 01), this pin outputs internal **MCLK_DATA** (output).
GPIO[13:14]	I/O	136, 137	These two pins always provide general purpose I/O function.
GNT2#/ nAutoFd	O	153	If parallel port is enabled (port 0x3e[4] = 1x), this pin serves as ECP "**nAutoFd**". If parallel port is not enabled (port 0x3e[4:5] = 0x), this pin outputs PCI bridge Grant Two "**GNT2#**".
GNT3#/ NStrobe	O	154	If parallel port is enabled (port 0x3e[4] = 1x), this pin serves as ECP "**nStrobe**". If parallel port is not enabled (port 0x3e[4:5] = 0x), this pin outputs PCI bridge Grant Three "**GNT3#**".
REQ2#/ NAck	I	155	If parallel port is enabled (port 0x3e[4] = 1x), this pin serves as ECP "**nAck**". If parallel port is not enabled (port 0x3e[4:5] = 0x), this pin inputs master Request Two "**REQ2#**".
REQ3#/ Busy	I	156	If parallel port is enabled (port 0x3e[4] = 1x), this pin serves as ECP "**Busy**". If parallel port is not enabled (port 0x3e[4:5] = 0x), this pin inputs master Request Three "**REQ3#**".
PCI/WIO Bus Bridge			
INTD#/ **WGLBCS#**	O	32	During **PCI** cycles: If WIO is enabled, this signal shall not connect to any PCI bus master. During **WIO** cycles: Asserted low indicating a WIO command cycle is ongoing
PDA[31:24]/ **WA[23:16]/** **WD[15:8]**	I/O	100-98, 94, 93, 90-88	During **PCI** cycles: These pins serve as the highest byte of PCI 32-bit address/data bus. During **WIO memory** cycles0: These pins serve as the highest byte of 24-bit address lines (XA[8:31]) During **WIO IO** cycles: These pins serve as the high byte of 16-bit data lines (XD[15:0]).
PDA[23:8]/ **WA[15:0]**	I/O	85, 84, 83, 82, 80-77, 63-61, 59-57, 50, 49	During **PCI** cycles: These pins serve as bits 16-31 of PCI 32-bit address/data bus. During **WIO** cycles: These pins serve as the lower 16-bit of 24-bit address lines (XA[8:31]).
PDA[7:0]/ **WD[7:0]**	I/O	45-38	During **PCI** cycles: These pins serve as the lowest byte of PCI 32-bit address/data bus. During **WIO memory** cycles: These pins serve as the 8-bit data lines. During **WIO IO** cycles: These pins serve as the low byte of 16-bit data lines (XD[15:0]).
C/BE3#/ **WIOCS#**	I/O	87	During **PCI** cycles: Bit-3 of Command/Byte Enable bus During **WIO** cycles: WIO chip-select for its IO devices
C/BE2#/ **WROMCS#**	I/O	76	During **PCI** cycles: Bit-2 of Command/Byte Enable bus During **WIO** cycles: WIO chip-select for its memory devices
C/BE1#/ **WWR#**	I/O	65	During **PCI** cycles: Bit-1 of Command/Byte Enable bus During **WIO** cycles: Asserted low, if WGLBCS# is low also, indicating that a WIO write command cycle is ongoing
C/BE0#/ **WRD#**	I/O	47	During **PCI** cycles: Bit-0 of Command/Byte Enable bus During **WIO** cycles: Asserted low, if WGLBCS# is low also, indicating that a WIO read command cycle is ongoing

INTA#, INTB#, INTC#	I	36, 35, 33	PCI Interrupt input, level sensitive, low active signal. Once the INTx# signal is asserted, it remains asserted until the device driver clear the pending request. When the request is cleared, the device de-asserts its INTx# signal.
REQ0#, REQ1#	I	102, 101	PCI Request input, indicates to the PCI arbiter that this agent desires use of the bus.
GNT0# GNT1#	O	104, 103	PCI Grant output, indicates to the agent that access to the bus has been granted.
PCIRST#	O	23	PCI Reset output, is used to bring PCI-specific registers, sequencers, and signals to a consistent state. Low active.
PCICLK	O	21	PCI Clock output, provides timing for all transactions on PCI and is an input to every PCI device.
SERR#	I	67	PCI System Error is for reporting address parity errors, data parity errors on the Special Cycle command, or any other system error where the result will be catastrophic. The assertion of SERR# is synchronous to the clock and meets the setup and hold times of all bused signals.
PERR#	I/O	68	PCI Parity Error is only for the reporting of data parity errors during all PCI transactions except a Special Cycle. The PERR# pin is sustained tri-state and must be driven active by the agent receiving data two clocks following the data when a data parity error is detected. The minimum duration of PERR# is one clock for each data phase that a data parity error is detected. An agent cannot report a PERR# until it has claimed the access by asserting DEVSEL# (for a target) and completed a data phase or is the master of the current transaction.
STOP#	I/O	69	PCI Stop indicates the current target is requesting the master to stop the current transaction.
TRDY#	I/O	71	PCI Target Ready indicates the selected device ability to complete the current data phase of the transaction. A data phase is completed on any clock both TRDY# and IRDY# are sampled asserted. During a read, TRDY# indicates that valid data is present on PDA[31:0]. During a write, it indicates the target is prepared to accept data. Wait cycles are inserted until both IRDY# and TRDY# are asserted together.
DEVSEL#	I/O	70	PCI Device Select, when actively driven, indicates the driving device has decoded its address as the target of the current access. As an input, DEVSEL# indicates whether any device on the bus has been selected.
FRAME#	I/O	74	PCI Cycle Frame is driven by the current master to indicate the beginning and duration of an access. FRAME# is asserted to indicate a bus transaction is beginning. While FRAME# is asserted, data transfers continue. When FRAM# is de-asserted, the transaction is in the final data phase or has completed.
IRDY#	I/O	72	PCI Initiator Ready indicates the bus master ability to complete the current data phase of the transaction. A data phase is completed on any clock both IRDY# and TRDY# are sampled asserted. During a write, IRDY# indicates that valid data is present on PDA[31:0]. During a read, it indicates the master is prepared to accept data. Wait cycles are inserted until both IRDY# and TRDY# are asserted together.

PPAR	I/O	66	PCI Parity is even parity across PDA[31:0] and C/BE[3:0]#. PPAR is stable and valid one clock after the address phase. For data phases, PPAR is stable and valid one clock after either IRDY# is asserted on a write transaction or TRDY# is asserted on a read transaction. (PPAR has the same timing as PDA[31:0], but it is delayed by one clock.) The mater drives PPAR for address and write data phases; the target drives PPAR for read data phase.
8-bit Video-In (VMI) Bus			
VIN[0:7]	I/O	114~121	During **normal mode**, this bus **inputs** 8-bit digital components of YCbCr 4:2:2 video-in data from external video controller (ex. TV decoder or MPEG decoder). During **test mode**, this bus **outputs** 8-bit digital components of YCbCr 4:2:2 video-out generated by internal video accelerator (VA).
HIS	I	113	Horizontal Sync of video-in frames. The content of VPC[10] (port address BA + 0x100) determines the polarity of this signal.
VSI	I	112	Vertical Sync of video-in frames. The content of VPC[11] determines the polarity of this signal.
VCLK	I	110	This clock source serves as VMI bus pixel clock (27MHz). A precise **27MHz** clock source shall be connected to this pin.
Display and DAC interface			
CP2/C/Blue	AO	128	During **composite** video mode (NTSC, PAL), this pin is analog "composite video" output. During **S-Video** mode, this pin is analog "**chrominance**" output. During **monitor** mode, this analog output supplies current corresponding to the "**blue**" intensity of the pixel being displayed. (To maintain IBM VGA compatibility, R-G-B outputs are typically terminated to monitor's ground with a 75 ohms 2% resistor. This resistor, in parallel with the 75 ohms resistor in the monitor, will yield 37.5 ohms impedance to ground. For a full-scale voltage of 700 mV, full-scale current output will be **18.7mA**.)
CP1/Y/Green	AO	129	During **composite** video mode (NTSC, PAL), this pin is analog "composite video" output. During **S-Video** mode, this pin is analog "**luminance**" output. During **monitor** mode, this analog output supplies current corresponding to the "**green**" intensity of the pixel being displayed.
CP0/Red	AO	130	During **composite** video mode (NTSC, PAL), this pin is analog "composite video" output. During **S-Video** mode, this pin left no connection. During **monitor** mode, this analog output supplies current corresponding to the "**red**" intensity of the pixel being displayed.
HSO	O	134	Horizontal Sync of the displayed graphic output. The content of VAconf[?] determines the polarity of this signal.
VSO	O	132	Vertical Sync of the displayed graphic output. The content of VAconf[?] determines the polarity of this signal.

VREF	AO	125	Voltage Reference Output. Bypass and decouple the voltage reference with a 0.1uF ceramic capacitor to the VDDa. The decoupling capacitor shall be as close to this pin as possible. This pin as well as "COMP" is used to control the current of internal current sources to be exactly equal to "Iref".
COMP	AO	127	Compensation pin. It shall be decoupled with a 0.1uF ceramic capacitor to VDDa. The decoupling capacitor shall be as close to this pin as possible.
RSET	AO	126	Current Source Adjusting Resistor. This pin is used to adjust the full-scale current of TV's analog outputs. A resistor shall be connected between this pin and VSSa. The **Iref** is approximate to **1.16V/RSET** (The current mirror of DAC's "Iref" is adjusted by this pin).
EXTVREF	AI	123	External Vref input. This signal supplies the DAC's "bandgap" output from an external 1.235V voltage source. A 0.1uF bypass capacitor shall be always connected between this pin and VDDa. (The "bandgap" is an voltage stabilizer of voltage-reference-generator "Vref"). This pin may be left unconnected.
Memory Controller Interface			
CS0L#/RAS0#, CS1L#/RAS1#	O	195, 197	During **EDO mode**, these signals are served as RAS0#, RAS1# that used to latch the row address MA[0:11] lines into the DRAM. Each signal is used to select one DRAM bank. During **SDRAM mode**, these signals are served as CS0L#, CS1L# that indicates the command decoder is enable or disable.
CS0H#, CS1H#	O	201, 203	Similar to CS0L#/CS1L#, when on-board SDRAM is used, These pins are NC pins. When SDRAM DIMM module is used, these signals indicates current cycle accessing the high word (32 bits) of DIMM's data bus (double words).
RAS#, CAS#	O	205, 206	This signals along with WE# and CS# define the command code of SDRAM configuration cycles.
WE#	O	204	This signal asserted to indicate a write cycle to DRAM.
CAS[0:3]#/ DQMB[0:3]	O	6, 10, 16, 25	In **EDO mode**: these signals are served as CAS# function and used to latch the column address (MA[0:11]) into DRAMs. It also indicates which bytes can be accessed. In **SDRAM mode**: these signals are served as DQMB function, these are input mask signals for write cycle and output enable signals for read cycle.
MA[0:13]	O	3, 4, 5, 11, 12, 13, 14, 15, 26, 27, 28, 29, 30, 31	These signals are used to provide the multiplexed row and column address to the EDO DRAM or SDRAM.
MD[0:31]	I/O	157-162, 164, 166-173, 175, 177-184, 186, 188-194	These signals are used to interface to the DRAM data bus.
CKE	O	17	This signal is used to enable or disable MCLK into SDRAM.
MCLK	O	19	This signal is SDRAM clock input, all SDRAM input /output signals are referenced with MCLK rising edge.

COM0 Serial Port			
SIN0	I	207	COM0 serial data input from the communication link (modem or peripheral device).
SOUT0	O	1	COM0 serial data output to the communication link (modem or peripheral device).
CTS0n	I	56	COM0 clear to send signal
DSR0n	I	55	COM0 data set ready
DTR0n	O	51	COM0 data terminal ready
RTS0n	O	52	COM0 request to send
DCD0n	I	53	COM0 data carrier detect
RI0n	I	54	COM0 ring indicator
COM1 Serial Port			
SIN1	I	208	COM1 serial data input from the communication link (modem or peripheral device).
SOUT1	O	2	COM1 serial data output to the communication link (modem or peripheral device).
Synchronous Serial Port			
SDI	I	106	Serial data-in from a external codec device
SDO	O	105	Serial data-out to a external codec device
SYNC	I/O	107	Frame sync of SDI/SDO. This signal is an input signal during "slave mode" (CFGH[2] set low)or output signal during "master mode" (CFGH[2] set high)
SCLK	I/O	108	Serial Clock for SDI/SDO transferring. This signal is an input signal if **MD[25]** is pulled **low**, and act as output signal during is MD[25] is pull **high**.
Miscellaneous :			
INTR0	I	91	This pin serves as an external interrupt request. A active high-state in this pin will make EIER[12] be set. This pin may also serve as an interrupt request pin for an IDE slot.
INTR1	I	92	This pin serves as an external interrupt request. A active high-state in this pin will make EIER[13] be set. This pin may also serve as an interrupt request pin for an IDE slot.
Power/Ground pin :			
VDD5V	I	24, 75, 122	5.0V Vdd (for a mixed 5.0V/3.3V environment)
VDDp	I	9, 22, 37, 48, 64, 81, 95, 109, 133, 147, 163, 174, 185, 202	Global 3.3V Vdd
VSSp	I	7, 18, 20, 34, 46, 60, 73, 84, 97, 111, 135, 152, 165, 176, 187, 198	Global VSS
VDDa	I	131	3.3V Vdd for DAC
VSSa	I	124	VSS for DAC
VDDl	I	196	3.3V Vdd for PLL
VSSl	I	200	VSS for PLL

5 MEGACELLS

5.1 Clock Module and PLL

5.1.1 Block Diagram

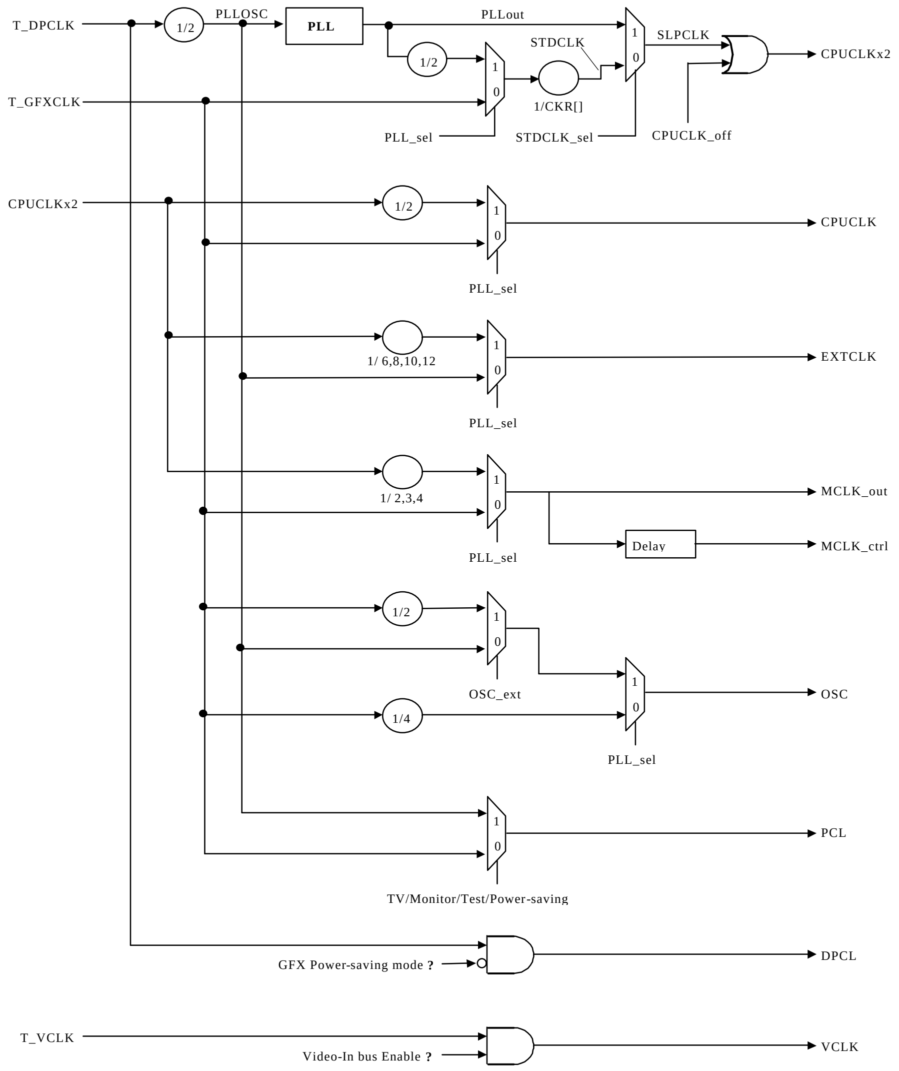


Figure 5.1.1 Block diagram of clock generation circuit.

5.1.2 Features

□ Single **27MHz** oscillator to generate all clock sources

- **PLL input clock (13.5MHz)**
- **CPU clock range : 80 MHz, 90 MHz, 100 MHz**

- **Video-Accelerator clock (27MHz, 13.5MHz)**
- **system clock (CPU-clock/2, /3, /4)**
- **SDRAM clock (CPU-clock/1, /1.5, /2)**
- **UART, Timer clock (13.5MHz)**

□ Optional SCLK input for SSI interface as SSI operation in slave mode

5.2 Power-on Settings

MD Bit(s)	Value	Setting
Status of Internal PLL		
MD[31]	0	Disable
	1	Enable
Speed of Internal System Clock Signal (CPUCLK)		
MD[28:30]	111	80 MHz
	110	90 MHz
	101	100 MHz
Speed of Internal PCI Clock Signal (EXTCLK)		
MD[26:27]	00	CPUCLK / 3
	01	CPUCLK / 4
	10	CPUCLK / 5
	11	CPUCLK / 6
Mode of Synchronous Serial Interface		
MD[25]	0	Slave
	1	Master
Speed of internal OSC Clock Signal		
MD[24]	0	DPCLK / 2
	1	GFXCLK / 2
Reserved for Future Use		
MD[20:23]	NA	NA

MD[**31**] pulled high to enable internal PLL unit

MD[**28-30**] used to choose CPUCLK from 80MHz to 110MHz

[1, 1, 1]:	80 MHz
[1, 1, 0]:	90 MHz
[1, 0, 1]:	100 MHz
[1, 0, 0]:	110 MHz

MD[**26-27**] used to choose EXTCLK (PCICLK)

DX3 Mode:	2'b00 to choose 1/3 CPUCLK,
DX4 Mode:	2'b01 to choose 1/4 CPUCLK,
DX5 Mode:	2'b10 to choose 1/5 CPUCLK,
DX6 Mode:	2'b10 to choose 1/6 CPUCLK

MD[**25**] pulled high to set X_SCLK (of SSI unit) to output mode (SCLK master mode)

MD[**24**] set high to choose OSC clock as GFXCLK/2

MD[**20-23**] reserved for future use

CTM[0-3] (0xf00001d8) memorize states of **MD[20-23]** during power-on interval (Firmware reads these bits to know what kind of target board is operating.)

DRAMTCtrl2[2-3] (0xf000003e) defines MCLK frequencies

MX1 Mode:	2'b00 to choose 1/3 CPUCLK,
MX1.5 Mode:	2'b01 to choose 1/4 CPUCLK,
MX2 Mode:	2'b10 to choose 1/5 CPUCLK,

TVTWH[12-15] (0xf0000178) defines the format of video output

VPTC[17] (0xf000017c) defines the direction of 8-bit video-in bus

5.2.1 Related Pins

DPCLK (input)

This clock source serves as internal PLL input as well as VA's system clock. A precise 27MHz clock source shall be connected to this pin during normal operation.

VCLK (input)

This clock source serves as VMI bus pixel clock (27MHz). A precise 27MHz clock source shall be connected to this pin.

GFXCLK (input)

This clock source serves as pixel clock, 36MHz to 50MHz, using in 800x600 non-interlace monitor. For TV system, this clock may pull high or low externally. Meanwhile, FXCLK may also serve as system OSC (for baud rate or timer adjustment), if MD[24] is pull high externally.

SCLK (in/out)

This pin is the serial bit clock between SSI and codec device. The SCLK may be input or output depending on whether SSI is operated in slave- or master-mode.

VDDI, VSSI:

Dedicated power/ground pins for internal PLL unit. VDDI shall be connected to a 3.3V voltage source.

5.3 Operation Modes

□ Normal Mode (PLL is enable)

- **TV system**

  DPCLK = 27MHz
  VCLK = 27MHz
  GFXCLK may be unconnected
  PCLK = DPCLK/2 = 13.5MHz
  PLLin = DPCLK/2 = 13.5MHz
  CPUCLK = 80 ~ 100MHz
  EXTCLK = CPUCLK/3, CPUCLK/4, CPUCLK/5, CPUCLK/6
  MCLK = CPUCLK/1, CPUCLK/1.5, CPUCLK/2
  OSC = DPCLK/2 = 13.5MHz (MD[24] = 0 on power-on reset)
  = GFXCLK/2 = 18.432MHz (MD[24] = 1 on power-on reset)

- **Monitor**

  DPCLK = 27MHz
  VCLK = 27MHz
  GFXCLK = **36.864MHz** (for deriving 18.432MHz OSC)
  PCLK = GFXCLK = 36.864MHz
  PLLin = DPCLK/2 = 13.5MHz
  CPUCLK = 80 ~ 100MHz

EXTCLK = CPUCLK/3, CPUCLK/4, CPUCLK/5, CPUCLK/6
MCLK = CPUCLK/1, CPUCLK/1.5, CPUCLK/2
OSC = DPCLK/2 = 13.5MHz (MD[24] = 0 on power-on reset)
= GFXCLK/2 = 18.432MHz (MD[24] = 1 on power-on reset)

□ Test Mode (PLL is disable)
DPCLK = 27MHz
VCLK = 27MHz
GFXCLK = 66MHz
PCLK = DPCLK/2 = 13.5MHz
CPUCLK = GFXCLK = 66MHz (for instance)
EXTCLK = CPUCLK / 2 = 33MHz
MCLK = CPUCLK = 66MHz
OSC = CPUCLK / 4 = 16.5MHz

5.4 PA-RISC CPU core

5.4.1 Features

□ Base on PA-RISC 1.1 level-0 architecture
□ 32-bit integer instruction set and register files
□ Maximum 100 MHz operation frequency
□ 3.3V and 0.01W/MHz at full speed operation
□ High-speed 32-bit integer pipeline design

- 6 stages for Load/Store instructions
- 5 stages for other instructions

□ On-chip cache memory

- 4 KB, direct-map instruction cache and 4 KB, 4-way set-associative data cache
- Write-through and write-back support for data cache
- One level read buffer and wrap-around support in each cache
- One level write buffer and hit-under-miss support in data cache
- Cache-locking support in instruction cache

□ Dynamic branch prediction

- Build-in 1-level 256 entry, 4-way set-associative (LRU) Branch-Target-Buffer to improve branch prediction rate and accelerate pipeline throughput

□ One high speed (2 CPU cycles) 16-/32-bit MAC and multimedia extended instructions have been built-in for DSP related calculation
□ Specific serial-ICE-interface to facilitate chip debugging and software development

5.4.2 Related Pins

PWRON (input)

System Power-On Reset signal; Set this signal to logic high will reset the chip and force all megacells returned to their initial states.

INTR0 (input)

This pin serves as an "external interrupt request". Set this signal to logic 1 will also set EIER[12] to logic 1. (The 16-bit IDE slot can use this pin as its interrupt request).

INTR1 (input)

This pin serves as another "external interrupt request". Set this signal to logic 1 will also set EIER[13] to logic 1. (The other 16-bit IDE slot can also use this pin as its interrupt request).

5.5 GPIO

5.5.1 Overview

The W90221X provides totally 19 GPIO pins. These pins may serves as traditional PIO, or parallel port interface, or another 2 PCI bus master request/grant, depend on what **bit[4:5**] of port **0xf000003e** are set. Right after power-on reset, all these pins are hi-Z (input mode).

5.5.2 Block Diagram

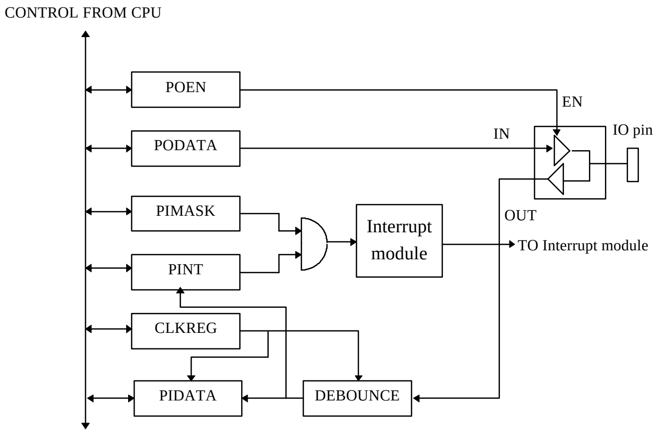


Figure 5.5.2 Block diagram of GPIO.

5.5.3 Features

- □ Each PIO port can generate a separate positive and negative edge interrupt.
- □ Each PIO port consists of a bi-directional buffer connected to the appropriate W90221X pin.
- □ The input buffer is routed directly to a de-bounce circuit.
- □ The de-bounce circuit performs a two TCLK_BUN clocks de-bounce of the input signal.
- □ Programmable de-bounce circuit sampling clocks (TCLK_BUN), the clock range is TCLK ~ TCLK/128.

5.5.4 Related Pins

Table 5.4-1 : GPIO definitions

# of GPIO	cfg = 00	cfg = 01	cfg = 1x
GPIO[0:7]	PIO[0:7] (io)	PIO[0:7] (io)	ED[0:7] (io)
GPIO[8]	PIO[8] (io)	PIO[8] (io)	nInit (out)

GPIO[9]	PIO[9] (io)	PIO[9] (io)	nSelectIn (out)
GPIO[10]	PIO[10] (io)	**CPUCLK** (out)	Select (In)
GPIO[11]	PIO[11] (io)	**MCLK_CTL** (out)	PError (In)
GPIO[12]	PIO[12] (io)	**MCLK_DATA** (out)	nFault (In)
GPIO[13]	PIO[13] (io)	PIO[13] (io)	PIO[13] (io)
GPIO[14]	PIO[14] (io)	PIO[14] (io)	PIO[14] (io)
GPIO[15]	**GNT2# (out)**	**GNT2# (out)**	nAutoFd **(out)**
GPIO[16]	**GNT3# (out)**	**GNT3# (out)**	nStrobe **(out)**
GPIO[17]	**REQ2# (In)**	**REQ2# (In)**	nAck **(In)**
GPIO[18]	**REQ3# (In)**	**REQ3# (In)**	Busy **(In)**

Note : cfg = bit[4:5] of port 0xf000003e

5.5.4 Operation Modes

The following figure shows PIO timing diagram for the input pin data -- the shortest time, the pio interrupt register will generate an interrupt and will latch the pin input data into the input data register.

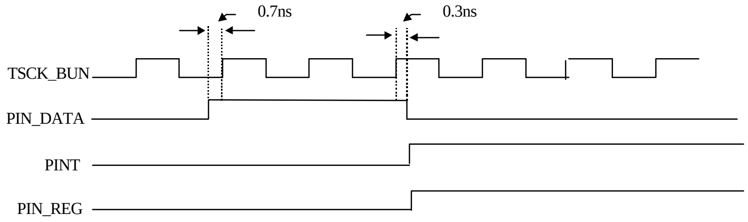


Figure 5.5.4.1 Shortest PIN_DATA should be kept to generate interrupt and latch the input data to PIO.

The following figure shows PIO timing diagram for the input pin data -- the longest time, the PIO interrupt register will not generate an interrupt and will not latch the pin input data into the input data register.

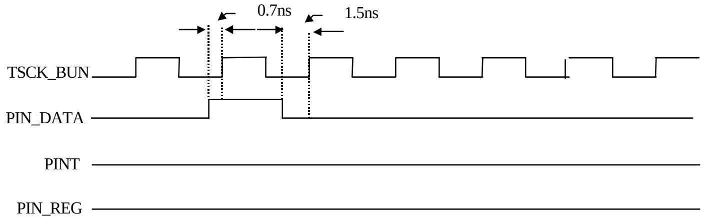


Figure 5.5.4.2 Longest PIN_DATA will not generate interrupt and will not latch the input data to PIO.

5.6 MEMORY CONTROLLER

5.6.1 Overview

The MEMC module within W90221X contains configuration register, control register, timing control registers and other logic to provide 32 bits SDRAM interface with external SDRAM memory devices, the flexible timing programming can achieve you use different speed of SDRAM whatever it is 32 bits on board.

5.6.2 Block Diagram

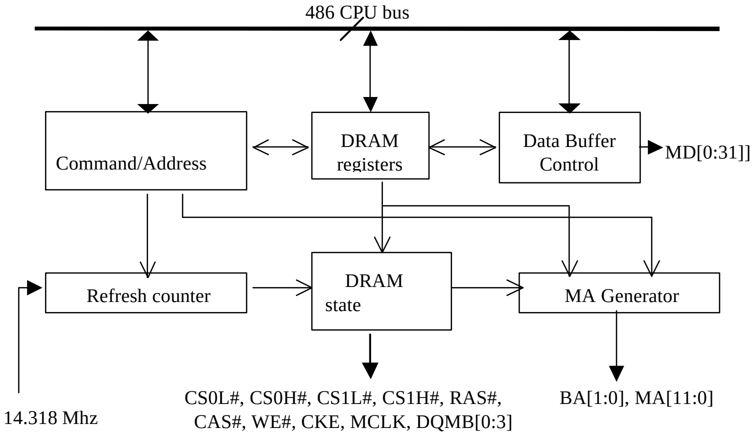


Figure 5.6.2.1 Block diagram of memory controller.

5.6.3 Features

- □ Supports up to 2 banks of SDRAM (DIMM or ON BOARD)
- □ 32-bit data interface
- □ CAS#-Before-RAS# refresh cycles for DRAM module
- □ Programmable RAS#/CAS# timing for DRAM access
- □ Supports only a burst length of one and burst type of sequential
- □ Programmable CAS# latency access time
- □ Provides 1M, 2M, 4M, 8M, 16M DRAM with page size 256 bytes, 512 bytes, 1K bytes, 2K bytes, 4K bytes configuration

5.6.4 Related Pins

CS0L#, CS1L# (out)

These signals are served as CS0L#, CS1L# In SDRAM mode that indicates whether the command decoder is enable or disable.

RAS#, CAS# (out)

This signals along with WE# and CS# define the command is being entered when using SDRAM configuration.

WE# (out)

This signals asserted indicates a write cycle to DRAM.

DQMB[0:3] (out)

These signals are served as DQMB function. These are input mask signals for write cycle and output enable signals for read cycle.

MA[0:13] (out)

These signals are used to provide the multiplexed row and column address to the SDRAM.

MD[0:31] (in/out)

These signals are used to interface to the DRAM data bus.

CKE (out)

This signal is used to enable or disable MCLK into SDRAM.

MCLK(out)

This signal is SDRAM clock input, all SDRAM input /output signals are referenced with MCLK rising edge.

5.6.5 Operation Modes

MX1 Mode

Once DRAMTctrl2[2:3] is set to 00, memory controller frequency is same as CPUCLK.

MX1.5 Mode

Once DRAMTctrl2[2:3] is set to 01, Memory controller frequency is CPUCLK/1.5.

MX2 Mode

Once DRAMTctrl2[2:3] is set to 10, Memory controller frequency is CPUCLK/2.

MCLK Skew Control

The SDRAM's CLK and internal MEMC system clocks are adjustable for SDRAM operating in higher clock rate (larger than 80 MHz). Three bit groups are used to define these clocks' skew, which are one SDRAM CLK and two internal MEMC system clocks. Following are some suggested setting when SDRAM operated in different modes: (Refer to DRAMctrl definition for details)

- MX1
  DRAMctrl[9:11] = 110
  DRAMctrl[12:14] = 010
  DRAMTctrl[9:11] = 001

- MX1.5
  DRAMctrl[9:11] = 010
  DRAMctrl[12:14] = 010
  DRAMTctrl[9:11] = 010

- MX2
  DRAMctrl[9:11] = 110
  DRAMctrl[12:14] = 001
  DRAMTctrl[9:11] = 110

5.6.5 Application Notes

The MEMC supports SDRAM while only 32-bit data bus is available. When using 64-bit SDRAM DIMM, special data bus routing is needed (for detail, refer to "Intel PC SDRAM Unbuffered DIMM Specification"). The following figures show typical connections and timing between W90221X and SDRAM:

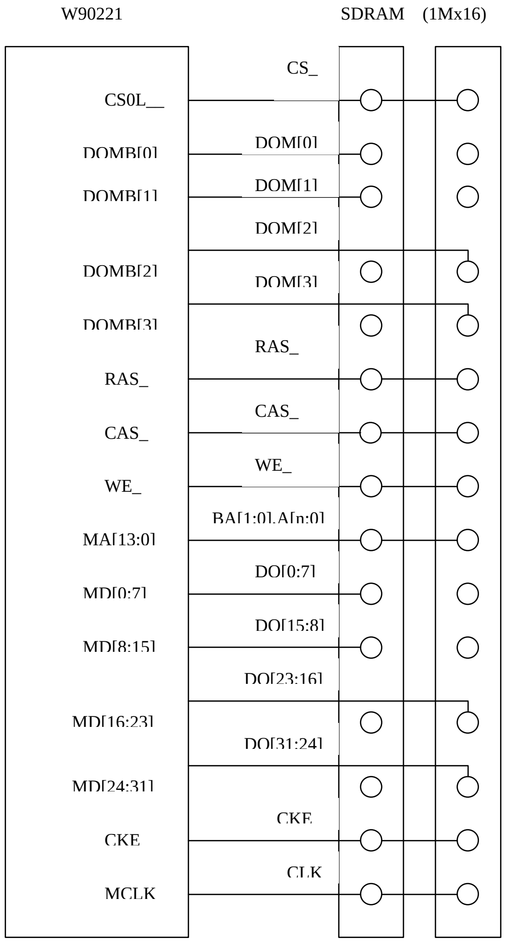


Figure 5.6.5.1 Onboard SDRAM connection

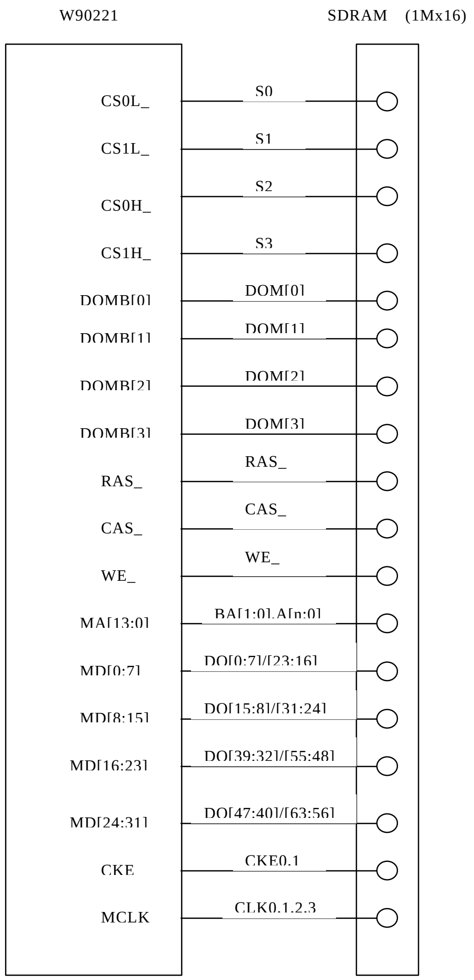


Figure 5.6.5.2 DIMM connection

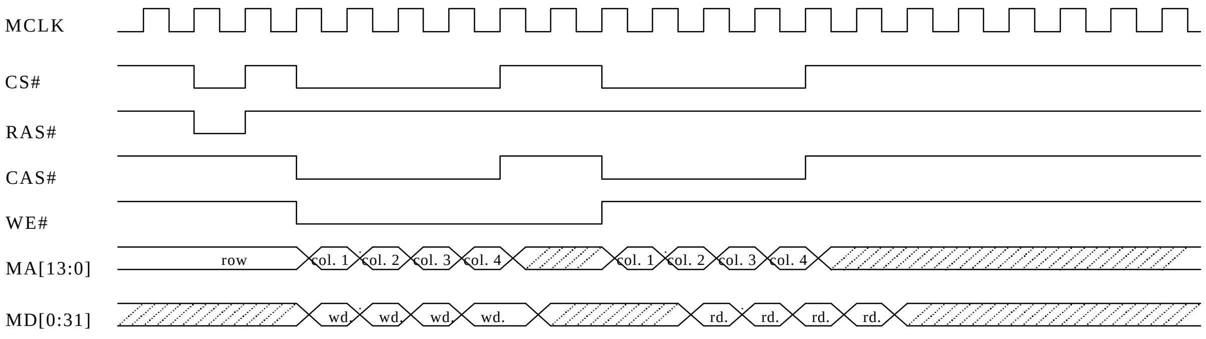


Figure 5.6.5.3 The fastest 32-bit SDRAM memory read/write cycle.

5.7 VIDEO ACCELERATOR (VA)

5.7.1 Overview

Display Controller controls the display timing and data to meet the requirements of display devices. The display controller of W90221X supports RGB monitor output and S-Video, RCA-style composite TV output. Pseudo color and high-color mode are used for graphics data. The pseudo color modes include 4-clor, 16-color and 256-color mode, while the high color mode is 565. Furthermore, W90221X supports the opaque function to save data bandwidth. This Display Controller includes the following modules to complete its functions: FIFO and Flicker-Free filter, Color Space Conversion, Overlay Control, Tune, CRTC, WIN-KEY, TV-Encoder

5.7.2 Block Diagram

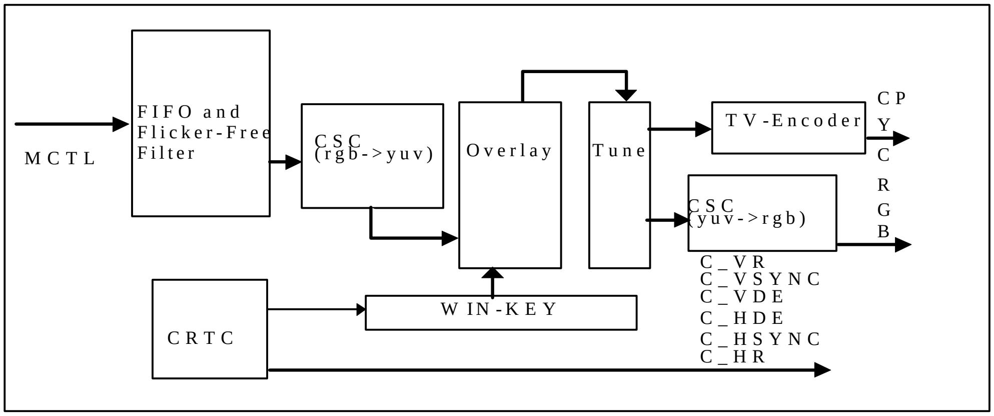


Figure 5.7.2.1 VA block diagram.

5.7.3 Features

□ Graphic Accelerator

- Build-in a hardware cursor with resolution up to 64x64x2

□ Video Accelerator

- Build-in buffers supporting YUV 4:2:2 video-in data
- Video-in may be 8-bit, 4 type of YUV component sequence are selectable for video-in
- Arbitrary-scaling-down and duplicated-scaling-up for video-in overlap in graphic display
- 2D bi-linear interpolation scaling-up for full-screen video display
- On chip CCCIR 601 YUV to RGB color space converter (CSC)

□ Video Overlay Logic

- Provide color key and window key
- Full screen display switch for graphics and video data

□ Display Interface

- Supports analog monitor up to 800x600 resolution, high color, non-interlace, 40MHz pixel clock and 60 frames/sec
- On chip TV encoder supporting NTSC, PAL or PAL-M system
- RCA-style composite video and S-Video
- Triple 8-bit RGB video DACs are integrated
- 3-line flicker free filter

5.7.4 Related Pins

□ Video-in interface

VIN[0:7] (in/out)

During normal mode, this bus inputs 8-bit digital components of YCbCr 4:2:2 video-in data from external video controller (e.g. TV decoder or MPEG decoder). During test mode, this bus outputs 8-bit digital components of YCbCr 4:2:2 video-out generated by internal video accelerator (VA).

HSI (input)

Horizontal sync of video-in frames: The content of VAconf[?] determines the polarity of this signal.

VSI (input)

Vertical Sync of video-in frames: The content of VAconf[?] determines the polarity of this signal.

VCLK (input)

As mentioned in section 5.1 "Clock Module and PLL"

□ Display interface

CP2/C/Blue (output)

During composite video mode (NTSC, PAL), this pin is analog "composite video" output. During S-Video mode, this pin is analog "chrominance" output. During monitor mode, this analog output supplies current corresponding to the "blue" intensity of the pixel being displayed. (To maintain IBM VGA compatibility, R-G-B outputs are typically terminated to monitor's ground with a 2% 75 ohms resistor. This resistor, in parallel with the 75 ohms resistor in the monitor, will yield 37.5 ohms impedance to ground. For a full-scale voltage of 700 mV, full-scale current output will be 18.7mA.)

CP1/Y/Green (output)

During composite video mode (NTSC, PAL), this pin is analog "composite video" output. During S-Video mode, this pin is analog "luminance" output. During monitor mode, this analog output supplies current corresponding to the "green"

CP0/Red (output)

During composite video mode (NTSC, PAL), this pin is analog "composite video" output. During S-Video mode, this pin left no connection. During monitor mode, this analog output supplies current corresponding to the "red" intensity of the pixel being displayed.

HSO (output)

Horizontal sync of the displayed graphic output: The content of VAconf[?] determines the polarity of this signal.

VSO (output)

Vertical sync of the displayed graphic output: The content of VAconf[?] determines the polarity of this signal.

□ DAC interface

VREF (analog Out) (optional)

Voltage Reference Out; Bypass and decouple the voltage reference with 0.1uF ceramic capacitor to the TVDD. The decoupling capacitor shall be as close to the chip as possible. This pin as well as "COMP" are used to control the current of internal current sources are exactly equal to "Iref".

COMP (analog Out)

Compensation pin: It shall be decoupled with a 0.1uF ceramic capacitor to TVDD. The decoupling capacitor shall be as close to the chip as possible.

RSET (analog Out)

Current source adjusting resistor: This pin is used to adjust the full-scale current of TV's analog outputs. A resistor shall be connected between this pin and TVSS. (The DAC's "Iref" current mirrors are adjusted by this pin). The Iref is approximate to 1.16V/RSET.

EXTVREF (analog In) (optional)

External Vref input. This signal supplies the DAC's "bandgap" output from a external 1.235V voltage source. A 0.1uF bypass capacitor should be always connected between this pin and TVDD. ("bandgap" is an voltage stabilizer of voltage-reference-generator "Vref").

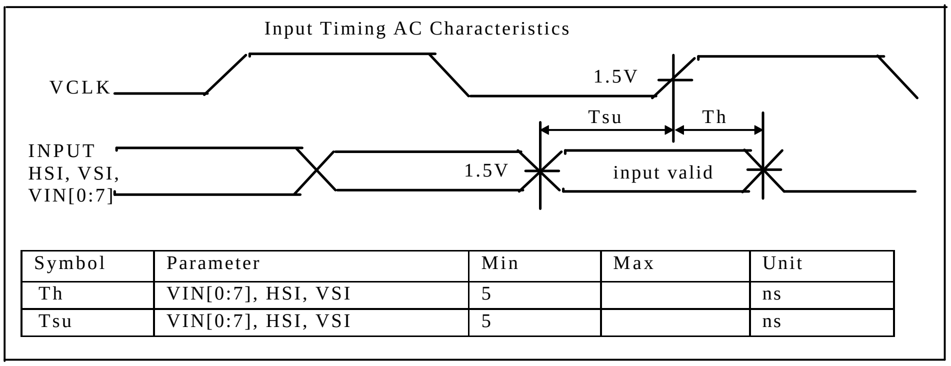
TVDD, TVSS

Dedicate power/ground pins for internal DACs.

5.7.5 Application Notes

A digital camera, or an NTSC/PAL camera connected to a TV decoder is fed into the W90221X in YCbCr 4:2:2 format through 8-bit data bus(VIN[0:7]). The input video is cropped and scaled, then display to output device. During operation mode, VCC[22] of VPRE must be set to zero to control data bus direction. And control data stream format by VCC[26:27] of VPRE.

The following figures show some typical timing diagrams of VPRE input pins:



Symbol	Parameter	Min	Max	Unit
Th	VIN[0:7], HSI, VSI	5		ns
Tsu	VIN[0:7], HSI, VSI	5		ns

Figure 5.7.5.1 VPRE timing diagram

The video pos-processing(VPOST) block that is designed to support two kind of display devices, TV and Monitor. On TV Mode, when TVTWH[12:13] is set to 1X, W90221X operate on TV-system. Clock (DPCLK) is equal to 27MHz. On Monitor Mode, when TVTWH[12:13] is set to 0X, W90221X operate on Monitor mode. Clock (GFXCLK) is equal to 36.864MHz.

5.8 DMA CONTROLLER

5.8.1 Features

- Flexible block-transfer mode and demand mode are supported
- Provides 8-bit ECP-to-memory or memory-to-ECP transfer mode
- Provides 8-, 16- and 32-bit memory-to-memory transfer modes
- DMA transfer between PCI memory to/from system memory are also support
- 4 words (16 bytes) memory burst-access; linear burst order
- Build-in 4-words data FIFO to accelerate memory access
- The starting address of source and target shall be half-word boundary for 16-bit memory transfer and word
- Boundary for 32-bit memory transfer

5.9 PCI BRIDGE

5.9.1 Overview

The W90221X host bridge provides a PCI bus interface that is compliant with the PCI local bus specification, revision 2.1. The implementation is optimized for high performance data streaming when the W90221X is acting as either the target or the initiator on the PCI bus.

5.9.2 Features

- Supports up to four external PCI bus masters
- Programmable external PCI clock signal (PCICLK) delay relative to internal PCI clock signal (EXTCLK)
- Provides fix/rotate arbitration algorithms
- Provides configuration read/write, I/O read/write, and memory read/write accesses

5.9.3 Relates Pins

PCLK (In)

PCLK provides timing for all transactions on PCI and is an input to every PCI device.

FRAME# (In/Out)

Cycle Frame, FRAME#, is an output when W90221X acts as an initiator on the PCI bus. FRAME# is driven by the current initiator and indicates the start (when it's first asserted) and duration (the duration of its assertion) of a transaction. While FRAME# is asserted, data transfer continue. When FRAME# is de-asserted, the transaction is in the final data phase or has completed. FRAME# is an input when W90221X acts as a PCI target.

PDA[31:0] (In/Out)

These signals are connected to the PCI address/data bus. Address is driven by W90221X with FRAME# being asserted; data is then driven or received in the following clocks. When W90221X acts as a target on the PCI bus, the AD[31:0] signals are inputs and contain the address during the first clock cycle of FRAME# assertion and input data(writes) or output data(reads) on subsequent clocks.

C/BE#[3:0] (In/Out)

PCI bus command and byte enable signals are multiplexed on the same lines. The Command/Byte Enable bus, C/BE#[3:0], defines the type of transaction during the address phase of a transaction. During the data phase, C/BE#[3:0] is used as byte enable signals. The byte enable signals determine which data paths carrying meaningful data. The PCI command types that are acknowledged by the W90221X are listed below:

C/BE3#	C/BE2#	C/BE1#	C/BE0#	Command Type
0	0	1	0	I/O Read
0	0	1	1	I/O Write
0	1	1	0	Memory Read
0	1	1	1	Memory Write
1	0	1	0	Configuration Read
1	0	1	1	Configuration Write

IRDY# (In/Out)

Initiator Ready, IRDY#, is an output when W90221X acts as an initiator on the PCI bus and an input when W90221X acts as a PCI target. During a write, IRDY# asserted indicates that the initiator is driving valid data onto the data bus. During a read, IRDY# asserted indicates that the initiator is ready to accept data from the current-addressed target.

TRDY# (In/Out)

Target Ready, TRDY#, is an input when W90221X acts as an initiator on the PCI bus and an output when W90221X acts as a PCI target. It is asserted when the target is ready to complete the current data phase (data transfer). A data phase is completed when the target is asserting TRDY# and the initiator is asserting IRDY# at the rising-edge of the PCICLK signal.

STOP# (In/Out)

STOP# is an input when W90221X acts as an initiator on the PCI bus and an output when W90221X acts as a PCI target. STOP# is used for disconnect, retry, and abort sequences on the PCI bus.

DEVSEL# (In/Out)

Device select, when asserted, indicates that a PCI target device has decoded its address as the target of the current access. The W90221X asserts DEVSEL# based on the DRAM address range being accessed by a PCI initiator. As an input it indicates whether any device on the bus has been selected.

PERR# (In/Out)

PERR# indicates the current transaction has data parity error occurs. It is an input when W90221X acts as an PCI initiator and the current transaction is write access or W90221X acts as an PCI target and the current transaction is read access. it is an output when W90221X acts as PCI initiator and the current transaction is read access or W90221X acts as an PCI target and the current transaction is write access. When PERR# asserted and Master 0 Latency Register bit 17=1, it will generate MI (non-maskable interrupt).

PPAR# (In/Out)

PPAR# is driven by the W90221X when it acts as a PCI initiator during address and data phases for a write cycle, and during the address phase for a read cycle. PPAR is driven by the W90221X when it acts as a PCI target during each data phase of a PCI memory read cycle. Even parity is generated across PDA[31:0] and COMBE#[3:0].

REQ0# (In)

Each PCI master has a pair of arbitration lines that connect it directly to the W902321X. When a master requires the use of the PCI bus, it REQ0# is the PCI bus request signal used as an input to indicate the arbiter that the master requires the use of the PCI bus.

REQ1# (In)

REQ1# is the PCI bus request signal used as an input to indicate the arbiter that this master 1 desires use of the bus.

SERR# (In)

SERR# is the system error reporting, if SERR# asserted and Master 0 Latency Register bit16 =1, it will generate a NMI (non-maskable interrupt).

INTA# (In)

PCI agent can utilize the PCI Interrupt request line A, INTA#, to generate request for service.

INTB# (In)

PCI agent can utilize the PCI Interrupt request line B, INTB#, to generate request for service.

INTC# (In)

PCI agent can utilize the PCI Interrupt request line C, INTC#, to generate request for service.

PCIRST# (output)

PCIRST# is used to reset PCI device.

GNT0# (output)

GNT0# is the PCI bus grant output signals generated by the internal PCI arbiter.

GNT1# (output)

GNT1# is the PCI bus grant output signals generated by the internal PCI arbiter.

GPIO[16:15] (output)

If ECP not enable, the GPIO[16:15] indicates PCI bus grant output GNT[3:2]#.

5.9.4 Operation Modes

DX3 Mode

Once MD[26:27] is set to 00 during power on reset, the PCICLK will operate at CPUCLK/3 frequency.

DX4 Mode

Once MD[26:27] is set to 01during power on reset, the PCICLK will operate at CPUCLK/4 frequency.

DX5 Mode

Once MD[26:27] is set to 10 during power on reset, the PCICLK will operate at CPUCLK/5 frequency.

DX6 Mode

Once MD[26:27] is set to 11 during power on reset, the PCICLK will operate at CPUCLK/6 frequency.

PCICLK skew control

Bit 17 ~ 20 of "Master 2 Latency Register" are used to adjust the skew of PCICLK so as to make all other PCI control signals get enough setup and hold time relative to PCICLK. Some typical value are suggested as follows:

For DX3	REG2[17:20] = 0100
For DX4	REG2[17:20] = 0111
For DX5	REG2[17:20] = 1001
For DX6	REG2[17:20] = 1011

5.10 WIO CONTROLLER

5.10.1 Features

□ **16M** ($2^{24} = 2^{4}$ X 2^{20}) addressing capacity for memory devices and **64K** ($2^{16} = 2^{6}$ X 2^{10}) for IO devices
□ 8-bit or 16-bit IO access
□ 8-bit memory write, 32-bit data-memory read and 4*32-bit code-ROM burst read
□ Memory space (ROM/Flash) are always non-cacheable except code-ROM
□ Provide no DMA transferring
□ Programmable command wait states, set-up and hold time for all access

5.10.2 Related Pins

WIO bus is an ISA-like bus and shares the existing 37 pins with PCI bus bridge. When WIO bus is enabled, only the PCI interrupt requests INTA#, INTB# and INTC# are available. The INTD# has been used as WIO's global chip select in that case.

PDA[31:24]/**WA[23:16]/WD[15:8]** (In/Out)
PCI cycles: Serve as highest byte of PCI 32-bit address/data bus.
WIO memory cycle: Serve as highest byte of 24-bit address lines (WA[23:0]) during WIO memory cycles.
WIO IO cycles: Serve as high byte of 16-bit data lines (WD[15:0]) during WIO IO cycles.

PDA[23:8]/**WA[15:0]** (In/Out)
PCI cycles: Serve as bits 16-31 of PCI 32-bit address/data bus.
WIO cycles: Serve as lower 16-bit of 24-bit address lines (WA[23:0]) during all WIO cycles.

PDA[7:0]/**WD[7:0]** (In/Out)
PCI cycles: Serve as lowest byte of PCI 32-bit address/data bus.
WIO memory cycles: Serve as the 8-bit data lines during WIO memory cycles.
WIO IO cycles: Serve as low byte of 16-bit data lines (WD[15:0]) during WIO IO cycles.

C/BE3#/**WIOCS#** (In/Out)
PCI cycles: Bit-3 of command/byte bus
WIO cycles: WIO chip-select for its IO devices

C/BE2#/**WROMCS#** (In/Out)
PCI cycles: Bit-2 of command/byte bus
WIO cycles: WIO chip-select for its memory devices

C/BE1#/**WWR#** (In/Out)
PCI cycles: Bit-1 of command/byte bus
WIO cycles: Asserted low, if INTD# is also low, indicating a WIO write command cycle is ongoing.

C/BE0#/**WRD#** (In/Out)
PCI cycles: Bit-0 of command/byte bus
WIO cycles: Asserted low, if INTD# is also low, indicating a WIO read

command cycle is ongoing.

NTD#/**WGLBCS#** (output)

PCI cycles:	If WIO is enable, this signal shall not connect to any PCI bus master.
WIO cycles:	Asserted low indicating a WIO command cycle is ongoing.

5.10.3 Application Notes

WIO bus is designed to connect ISA-like, low speed devices such as code-ROM, Flash and 8-/16-bit IO devices. The WIO controller itself is a PCI slave device, if the address and access type of any PCI cycle match the WIOBASE or XMBASE[8:15] of WIO, the WIO controller responds the DEVSEL# and TRDY# to PCI bridge and generate correspond WIO bus signals to WIO devices in the "data phase" of current PCI cycle.

XMBASE[7] will be set right from chip reset, all PCI cycles will be treated as WIO access and all code read (PCI cycles) will return data from WIO bus. After the memory (XMBASE) and IO (WIOBASE) have been properly configured, XMBASE[7] shall be set logic low immediately to avoid possible wrong response from WIO controller.

Because WROMCS#, WIOCS#, WRD# and WWR# share the same pins with COMBE[0:3] of PCI bus, they might toggle during any PCI cycles. It is necessary to OR these control signals with INTD#, which serves as "WIO global chip-select" dedicatedly, before they reaching the WIO devices.

The following figures show some typical timing diagrams of WIO command cycles:

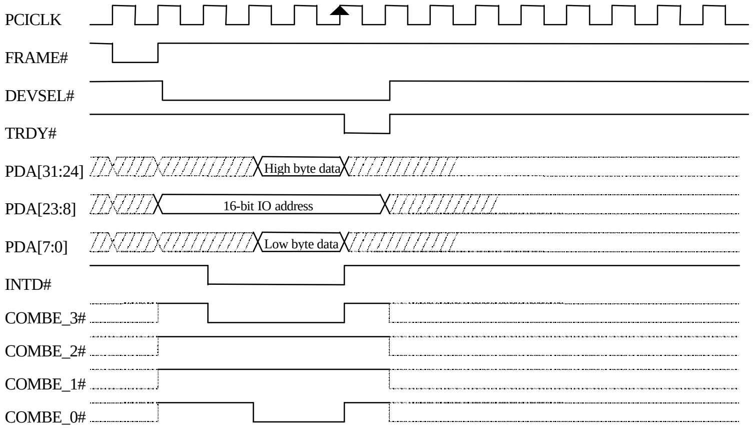


Figure 5.10.3.1 Fastest WIO I/O read cycle (0 wait)

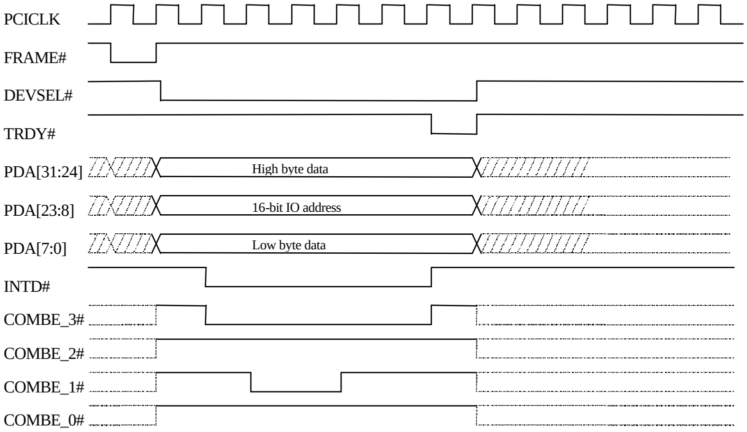


Figure 5.10.3.2 Fastest WIO I/O write cycle (0 wait)

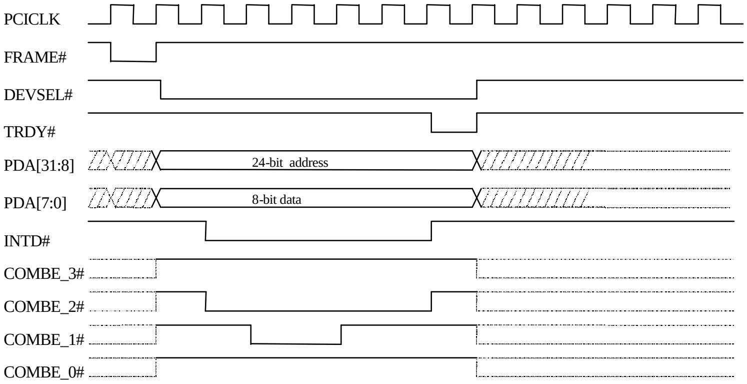


Figure 5.10.3.3 Fastest WIO memory write cycle (0 wait)

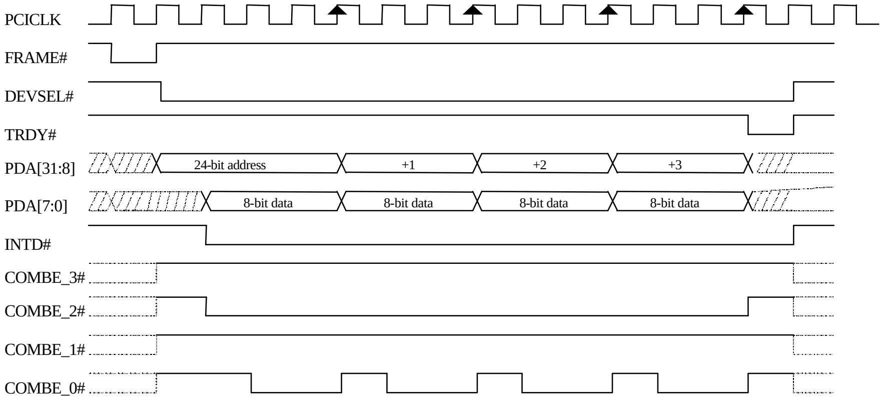


Figure 5.10.3.4 Fastest WIO memory write cycle (0 wait)

As for 4-word code ROM **burst read**, the timing diagram is similar to Fig5.8.4 except there will be consecutive **16** C/BE0## (WRD#) command pulses within one chip select (INTD#, C/BE2##).

5.11 PARALLEL PORT INTERFACE

5.11.1 Features

□ Supports all IEEE P1284 transfer modes including:

- Compatible (Centronic) mode (forward channel)
- Nibble mode (reverse channel, compatible with all existing PC hosts - relies on software control)
- Byte mode (reverse channel, compatible with IBM PS/2 host)
- EPP mode (bi-directional half-duplex channel - relies on software control)
- ECP mode (fast bi-directional half-duplex channel)

□ Host-side design

□ Provide a special operation mode to emulate peripheral-side Centronic device

□ Build-in one 16bytes FIFO to accelerate ECP mode and Centronic forward transfer

□ Provide DMA capability to accelerate moving data from parallel port interface to system memory

□ ECP mode is also including:

- High performance half-duplex forward and reverse channel
- Interlocked handshake, for fast reliable transfer
- Forward "channel-addressing/command transfer" for low-cost peripherals
- Support reverse RLE decompression
- Peer-to-peer capability

5.11.2 Related Pins

All 17 PPI interface signals share pins with GPIOs. When T_ENECP (port 0xf000003e, bit 4) is set, the corresponding GPIOs are redefined as follows:

GPIO[16]/**nStrobe** (output)
Compatible Mode: Set active low to transfer data into peripheral device's input latch
ECP Mode: Used in a closed-loop handshake with "Busy" to transfer data or address information from host to peripheral device.

GPIO[15]/**nAutoFd** (output)
Compatible Mode:Set low by host to put some printers into auto-line feed mode. May also be used as a ninth data, parity, or command/data control bit.
ECP Mode: The host drives this signal for flow control in the reverse direction. It is used in an interlocked handshake with "nAck". "nAutoFd" also provides a ninth data bit used to determine whether command or data information is present on the data signals in the forward transferring.

GPIO[8]/**nInit** (output)
Compatible Mode:Pulsed low in conjunction with "nSelectIn" active low to reset the interface and force a return to compatible mode idle state.
ECP Mode: This signal is driven low to place the channel in the reverse direction. While in this mode, the peripheral is only allowed to drive the bi-directional data signals when "nInit" is low and "nSelectIn" is high.

GPIO[9]/**nSelectIn** (output)

Compatible Mode: Set low by host to select peripheral device.

ECP Mode: Driven high by host while in ECP mode. Set low by host to terminate ECP mode and return the link to the compatible mode.

GPIO[17]/**nAck** (input)

Compatible Mode: Pulse low by the peripheral device to acknowledge transfer of a data byte from the host.

ECP Mode: Used in a close-loop handshake with "nAufoFd" to transfer data during reverse transferring.

GPIO[18]/**Busy** (input)

Compatible Mode: Driven high to indicate that the peripheral device is not ready to receive data.

ECP Mode: The peripheral device uses this signal for flow control in the forward transferring. "Busy" also provides a ninth data bit used to determine whether command or data information is present on the data signals in the reverse direction.

GPIO[11]/**PError** (input)

Compatible Mode: Driven high to indicate that the peripheral device has encountered an error in its paper path (ex. paper empty). Peripherals shall set nFault" low whenever they set "PError" high.

ECP Mode: Peripherals drive this signal low to acknowledge "nInit". The host relies upon "PError" to determine when it is permitted to drive the data signals.

GPIO[10]/**Select** (input)

Compatible Mode: Set high to indicate that the peripheral device is on-line.

ECP Mode: Used by peripheral to reply to the requested extensibility byte sent by the host during the negotiation phase.

GPIO[12]/**nFault** (input)

Compatible Mode: Set low by peripheral device to indicate that an error has occurred.

ECP Mode: Set high to acknowledge 1284 compatibility during negotiation phase. During ECP mode the peripheral may drive this pin low to request communications with the host. This signal would be typically used to generate an interrupt to the host. This signal is valid in both forward and reverse transfers.

GPIO[0:7]/**ED[0:7]** (in/out)

8-bit bus used to hold data, address or command information in all modes. The bit 0 is the most significant bit.

5.12 UART

5.12.1 Overview

The W90221X contains two Universal Asynchronous Receiver/Transmitter (UART) ports, one of them provides complete MODEM-control and serial transformation capabilities, whereas the other one provides only serial transformation capability. The UART performs serial-to-parallel conversion on data characters received from a peripheral device such as MODEM, and parallel-to-serial conversion on data characters received from the CPU. One 16 bytes transmitter FIFO (TX-FIFO) and one 16 bytes (plus 3 bits of error data per byte) receiver FIFO (RX-FIFO) have been built in to reduce the number of interrupts presented to the CPU. The CPU can read the complete status of the UART at any time during the functional operation.

Status reported includes error conditions (parity, overrun, framing, or break interrupt) and states of TX-FIFO and RX-FIFO.

5.12.2 Block Diagram

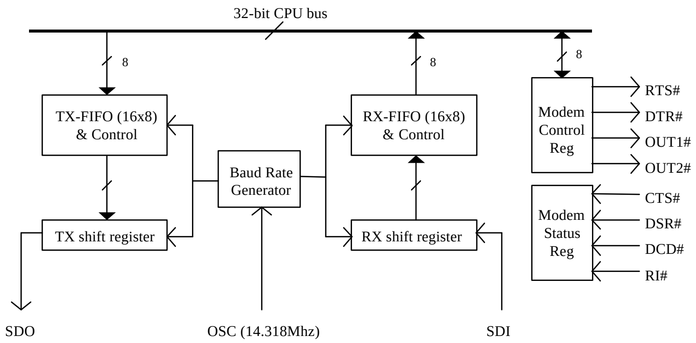


Figure 5.12.2.1 Block diagram of UART

5.12.3 Features

□ Transmitter and receiver are each buffered with 16 bytes FIFO's to reduce the number of interrupts presented to the CPU

□ MODEM control functions (CTS, RTS, DSR, DTR, RI and DCD)

□ Fully programmable serial-interface characteristics:

- 5-, 6-, 7-, or 8-bit characters
- Even, odd, or no-parity bit generation and detection
- 1-, 1&1/2, or 2-stop bit generation
- Baud rate generation

□ Line break generation and detection

□ False start bit detection

□ Full prioritize interrupt system controls

□ Loop back mode for internal diagnostic testing

5.12.4 Related Pins

□ **COM0**

SIN0 (input)
Serial data input from peripheral device or MODEM
SOUT0 (output)
Serial data output to peripheral device or MODEM
CTS0n (input)
Clear to send signal
DSR0n (output)
Data set ready
DTR0n (input)
Data terminal ready
RTS0n (output)
Request to send
DCD0n (input)
Data carrier detect
RI0n (output)

Ring indicator

□ **COM1**

SIN1 (input)

Serial data input from peripheral device or MODEM

SOUT1 (output)

Serial data output to peripheral device or MODEM

5.12.4 Operation Modes

□ Receiver control

- Set FCR[0:1] to select a proper receiver threshold level and then turn on "receiver data available interrupt" (Irpt_RDA) by set IER[7] to logic 1.
- The Irpt_RDA will be triggered when the receiver FIFO (RX-FIFO) has reached its programmed trigger level, and it will be cleared as the available data in RX-FIFO drops below the trigger level.
- As Irpt_RDA occured, the corresponding IIR bits will be set to inform the software application that data in RX-FIFO has reached programmed threshold level.
- If the received data has any errors, the "line status interrupt" (Irpt_RLS) will occur and has higher priority than Irpt_RDA.
- If "time out interrupt" (Irpt_TOR) is enable by set IER[7] and TOR[0] to logic 1s. The Irpt_TOR will occur, if the following conditions exist:
  - ✓ At least one character is in RX-FIFO
  - ✓ RX-FIFO is not received any data or accessed by CPU from the most recent serial character received, and the time period, counting by baud rate bit clock, has exceeded the value being programmed in TOR[1:7].
  - ✓ The Irpt_TOR and the time-out counter will be cleared as the CPU reads one character from RX-FIFO.
  - ✓ The time-out counter is reset after a new character is received or after the CPU reads the RX-FIFO.

□ Transmitter control

- Set IER[6] to logic 1 to enable "transmitter empty interrupt" (Irpt_THRE) before transmitter operation.
- Once the transmitter FIFO (TX-FIFO) is empty, the Irpt_THRE is triggered and the corresponding IIR bits are set to inform the CPU to fill the TX-FIFO (maximum 16 bytes of characters).
- The Irpt_THRE is reset after the CPU reads the IIR (IIR[4:7] must be 4'b0010 at that time) or writes a character into TX-FIFO.
- Irpt_RDA and Irpt_TOUT has the same interrupt priority (2nd priority) while Irpt_THRE has a lower priority (3rd priority).
- Polled Mode operation: (refer to "LSR" register descriptions located on Section 5.2.5)
- No interrupts need be enabled at this mode. The CPU always polls the LSR to check COM port status before taking any actions.
- LSR[7] will be set as long as there is at least one byte in the RX-FIFO, and it is cleared if the RX-FIFO is empty.
- LSR[3:6] will specify error(s) status which is handled the same way as in the interrupt mode operation, the IIR[4:7] is not affected since no interrupt is enabled.
- LSR[2] will indicate when the TX-FIFO is empty.

- LSR[1] will indicate that both TX-FIFO and shift register are empty.
- LSR[0] will indicate whether there are any errors in the RX-FIFO.

5.13 Synchronous Serial interface (SSI)

5.13.1 Overview

The SSI module within W90221X contains holding registers, shift registers, and other logic to support a variety of serial data communications protocols and provide a direct connection to external audio/telephony codec devices. Two 48-half-word FIFOs, one transmitter FIFO and one receiver FIFO, have been implemented to accelerate both transmitting and receiving operations. These two FIFOs can be configured as 48 half-word or 24 words depth depending on the data word length.

5.13.2 Block Diagram

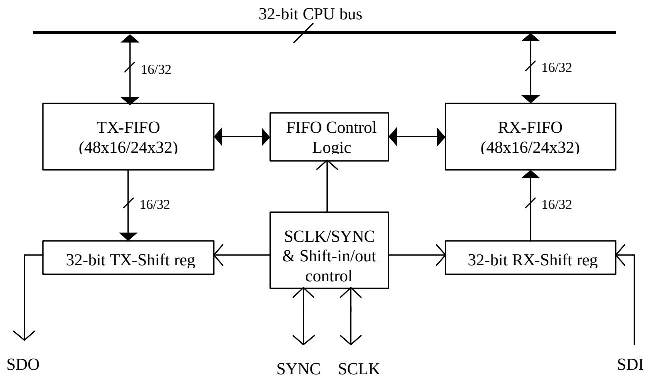


Figure 5.13.2.1 Block diagram of SSI

5.13.3 Features

- □ Supports "long framing" and "short framing" (synchronous, frame-based protocol)
- □ Provides "master mode" and "slave mode"
- □ Build-in two 48x16 (or 24x32) data FIFOs accelerating transmit/receive operation
- □ Programmable data bits per one frame (sampling rate) : 1 ~ 256 bits/frame
- □ Programmable data bits per word (resolution of each sampling) : 1 ~ 32 bits/word
- □ Programmable multi-word (per frame) transfer : 1 ~ 16 words/frame

5.13.4 Related Pins

SDI (input)

This pin contains the input data shifted from external audio/telephony codec devices.

SDO (output)

This pin contains the output data shifted to external audio/telephony codec devices.

SYNC (in/out)

This pin is the frame synchronization signal between SSI and codec devices. The SYNC may be input or output depending on SSI operated in slave- or master-mode respectively.

SCLK (in/out)

This pin is the serial bit clock between SSI and codec devices. Likewise, The SCLK may be input or output depending on SSI operated in slave- or master-mode respectively.

5.13.5 Operations Modes

Master Mode

Once CFGH[2] is set to logic 1 and MD[25] is pull high, SSI is operated in master mode, and the SYNC (determines the sampling rate) and SCLK are drove by SSI module to external CODEC devices.

SCLK frequency = EXTCLK/[2*(CFGL[8:15] + 1)]
SYNC period = SCLK * (CFGL[0:7] + 1)

Slave Mode

Once CFGH[2] is set to logic 0 and MD[25] is pull down, SSI is operated in slave mode, the SCLK and SYNC are drove externally (may be from CODEC devices). So the sampling rate and SCLK frequency are determined by external devices, however software driver still need to properly set "serial data bit length" (CFGH[8:10]) as well as "data words per frame" (CFGH[12:15]) to make SSI module working correctly.

Loop mode

This mode (CFGH[1] =1) aims at self-testing. When this bit is set, serial data-out "SDO" is connected to serial data-in "SDI" internally and SDO pin fixed at logic 0 state. Besides, if **Loop** and **Master** mode are chosen concurrently, SSI module will not issue SYNC until TX-FIFO contains at least one data word.

Long Framing

When CFGH[3] is set to logic 1, SSI is operated in long framing mode. The following features are included in long framing mode:

- The SSI module always samples receive date (SDI) on the falling edge of SCLK, whereas always pushes transmit data (SDO) on the rising edge of SCLK.
- The frame sync (SYNC) is asserted immediately as the first bit of transmit and receive data.
- The frame sync (SYNC) is asserted for one "serial word length" which determined by CFGH[8:11].

  Serial word length = CFGH[8:11] + 1
- The frame sync rate (sampling rate) and SCLK frequency follow equations (5.1.6b) and (5.1.6a) respectively on master mode and determined by external devices on slave mode.
- The transmit FIFO and receive FIFO is configured as 48x16 if "serial word length" <= 16, and will be configured as 24x32 if "serial word length" > 16.
- The shifting data bits on SDI and SDO are always MSB first.
- If serial word length is not 16 or 32, it is software responsibility to left(MSB) justify the transmit data words before writing it to transmit FIFO, the received data before being written into receive FIFO is right(LSB) justified automatically by SSI module where the unfilled MSBs are filled with logic 0s.
- SSI module always shifts out logic 0s on each frame sync if transmit FIFO is empty at that time.
- A receiver FIFO interrupt will be asserted (when RX-FIFO interrupt is enable) if the received data words exceeds the receive FIFO's threshold level. Likewise, a

transmitter FIFO interrupt will be asserted (when TX-FIFO interrupt is enable) if the available data words in transmit FIFO is lower than its threshold level.

- The following figure shows a standard long framing transfer where serial word length is 3 (CFGH[8:11] = 2), words per frame is 3 (CFGH[12:15]=2) and bits per frame is 9 (CFGL[0:7] = 10).

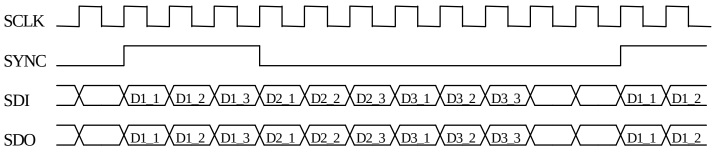


Figure 5.13.5.1 SSI long framing transfer

Short Framing

When CFGH[3] is set to logic 0, SSI is operated in long framing mode. The following features are included in short framing mode consists of the following features.

- The frame sync (SYNC) is asserted for one SCLK immediately before the first bit of transmit and receive data.
- The frame sync (SYNC) is asserted for one SCLK period.
- All other features are the same as long framing mode.
- The following figure shows a standard short framing transfer where serial word length is 3 (CFGH[8:11] = 2), words per frame is 3 (CFGH[12:15]=2) and bits per frame is 9 (CFGL[0:7] = 10).

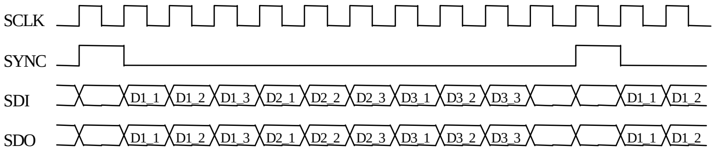


Figure 5.13.5.2 SSI short framing transfer

5.14 TIMER CHANNELS

5.14.1 Overview

Two 24-bit decrementing timers are implemented, corresponding to the TCR1, TICR1 and TCR2, TICR2 independently. When the timers' interrupt enable bit is set high and the counter decrements to zero, the timer will assert its interrupt request signal. When a timer reaches zero, the timer hardware reloads the counter with the value from the timer initial counter register and continues decrementing.

5.14.2 Block Diagram

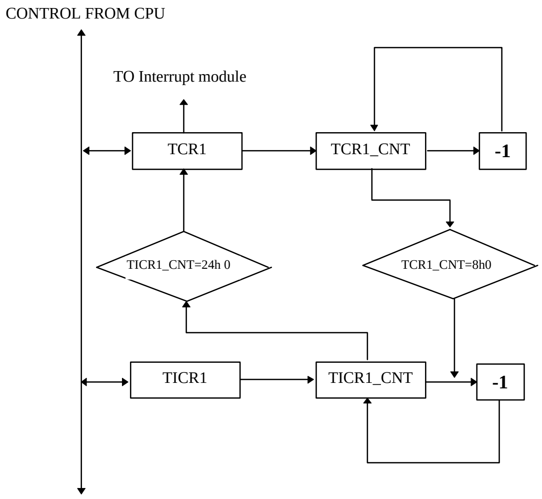


Figure 5.14.2.1 Block diagram of timer channel 1

Note:

The above block diagram is timer channel 1. Block diagrams of these two timer channels are the same.

5.14.3 Features

- □ Two 24-bit decrement timer channels with individual interrupt requests
- □ Programmable timer clocks for each channels, the clock range is OSC ~ OSC/8'hFF
- □ Maximum uninterrupted time or timeout = 5 minutes (if OSC = 14.318MHz)
- □ Typical OSC frequency is 14.318MHz.

5.14.4 Operation Modes

The following figures show some typical timing diagrams of TIMER write cycles

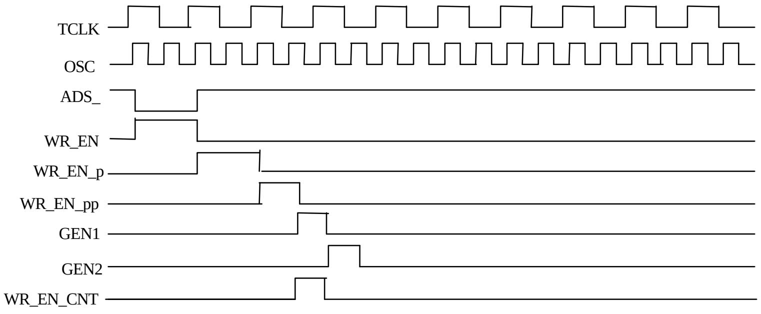
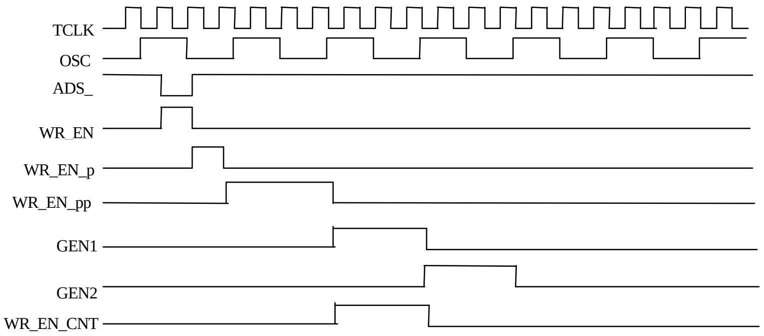


Figure 5.14.4.1 Timer register write command when the oscillator frequency is faster than the CPU frequency.



*Note :
WR_EN : The write enable signal for timer register reference the TCLK clock
WR_EN_CNT : The write enable signal for timer count register reference the OSC clock.

Figure 5.14.5.1 Timer register write command when the oscillator frequency is slower than the CPU frequency.

6 REGISTER DEFINITIONS

6.0 GPIO REGISTERS

There are six registers included in the GPIO module. The I/O address map is allocated from 0xf0000050 to 0xf0000064.

Port Address	Symbol	Access	Description
BA + 0x50	PIEN	R/W	Interrupt Enable Register
BA + 0x54	POEN	R/W	Output Enable Register
BA + 0x58	PODATA	R/W	Output Data Register
BA + 0x5C	PINT	R/W	Interrupt Request Register
BA + 0x60	PIDATA	R	Input Data Register
BA + 0x6C	CLKREG	R/W	Debounce Clock Select Register

GPIO Interrupt Enable Register (PIEN)

Port address: 0x00000050 **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
GPIO Interrupt Enable Mask [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
GPIO Interrupt Enable Mask [8:14]							Reserved

Bits 0-15 Reserved

Bits 16-30 GPIO interrupt enable mask bits

These fifteen bits serve as interrupt enable bits of GPIO[0:14] respectively, when the GPIOs are programmed as input mode. Setting these bits to logical 0s, the releated GPIO interrupt requests are pended in "PIO interrupt request" register (PINT).

Bit 31 Reserved

GPIO Output Enable Register (POEN)

Port address: 0x00000054 **Access type**: read/write **Default**: 0x0

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th></tr>
<tr><td colspan="8">GPIO Output Enable Mask [0:7]</td></tr>
<tr><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><td colspan="7">GPIO Output Enable Mask [8:14]</td><td>Reserved</td></tr>
</table>

Bits 0-15 Reserved

Bits 16-30 GPIO output enable mask bits

Setting any of these bits to logical high, then the corresponding GPIO[0:14] pins will act as output pins.

Bit 31 Reserved

GPIO Data-out Register (PODATA)

Port address: 0x00000058 **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Data-out [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Data-out [8:14]							Reserved

Bits 0-15	Reserved
Bits 16-30	GPIO data-out bits The logical state of these bits will be echoed on the corresponding GPIO[0:14] pins, if any of GPIO pins are programmed as output mode.
Bit 31	Reserved

GPIO Interrupt Request Register (PINT)

Port address: 0x0000005c **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
GPIO Interrupt Request [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
GPIO Interrupt Request [8:14]							Reserved

Bits 0-15 Reserved

Bits 16-30 GPIO interrupt request

When the GPIO is programmed as input pin, any transition in GPIO pins (a recognizable logic state shall keep stable for at least two debounce clocks (TCLK_BUN)) will set related bits in this register to logical high. Besides, the GPIO module will not issue interrupt request to CPU host unless the same bits of PIEN and PINT are both set high.

Bit 31 Reserved

GPIO Input Data Register (PIDATA)

Port address: 0x00000060 **Access type**: read **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Input Data [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Input Data [8:14]							Reserved

Bits 0-15 Reserved

Bits 16-30 GPIO pins status

These fifteen bits always echo GPIO[0:14] pin status no matter the GPIOs are programmed as input or output pins. All input ports of GPIO[0:14] are debunced first by TCLK_BUN before they are echoed by these bits.

Bit 31 Reserved

GPIO Input Data Register (PIDATA)

Port address: 0x00000060 **Access type**: read **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Input Data [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Input Data [8:14]							Reserved

Bits 0-15 Reserved

Bits 16-30 GPIO pins status

These fifteen bits always echo GPIO[0:14] pin status no matter the GPIOs are programmed as input or output pins. All input ports of GPIO[0:14] are debunced first by TCLK_BUN before they are echoed by these bits.

Bit 31 Reserved

Debounce Clock Select Register (CLKREG)

Port address: 0x00000064 **Access type**: read **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved					Debounce Clock Rate [0:2]		

Bits 0-28 Reserved

Bits 29-31 Debounce clock rate selector

These three bits are used to select the debounce circuit clock rate. The relationship of system clock (TCLK) and debounce clock (TCLK_BUN) according to these three bits are as follows:

TCLK_BUN = TCLK/CLKREG[29:31]

Each of fifteen general-purpose GPIO ports can be programmed as input or output port independently. Each port can generate positive or negative edge interrupt, and contains of a bi-directional buffer connected to the appropriate W90221X pin, the output signal from the input buffer is routed directly to a debounce circuit. This circuit performs 3 TCLK_BUN (program by CLKREG register) debounce of the input signal. Reading a specific bit location within the IO Data Input Register returns the logic state of the respective general purpose IO pin, regardless of whether that pin is configured as an output or input. If the pin is configured as an output an input, the value read is the logic state of the pin as driven by W90221X pin.

6.1 Memory controller Registers

There are five 16-bit registers and one 22-bit I/O base register included in the memory (DRAM) controller. Accessing to these registers is through an "I/O base address + offset" scheme. Accessing to the I/O base register (IOBASE) is through the 0xf0000000 port. The memory controller supports PC100-compliant synchronous DRAM.

Port Address	Symbol	Access	Description
BA + 0x30	DRAMCFG0	R/W	DRAM Bank 0 Configuration Register
BA + 0x32	DRAMBA0	R/W	DRAM Bank 0 Base Register
BA + 0x34	DRAMCFG1	R/W	DRAM Bank 1 Configuration Register
BA + 0x36	DRAMBA1	R/W	DRAM Bank 1 Base Register
BA + 0x38	DRAMCTRL	R/W	DRAM Control Register
BA + 0x3a	DRAMTCTL0	R/W	DRAM Timing Control Register 0
BA + 0x3c	DRAMTCTL1	R/W	DRAM Timing Control Register 1
BA + 0x3e	DRAMTCTL2	R/W	DRAM Timing Control Register 2
0xf0000000	IOBASE	R/W	I/O Base Address

DRAM Bank 0/1 Configuration Register (DRAMCFG0/DRAMCFG1)

Port address: 0x00000030/34 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
DRAM Size			DRAM Page Size			COMPBK	BANKEN
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							

Bits 0-2 Size of DRAM bank 0/1

DRAMCFG0/1 [0:2]	DRAM Size
000	1M
001	2M
010	4M
011	8M
100	16M
101	Reserved
110	Reserved
111	Reserved

Bits 3-5 DRAM page size

DRAMCFG0/1 [3:5]	Page Size (# of address lines)
000	256 Bytes (8)
001	512 Bytes (9)
010	1K Bytes (10)
011	2K Bytes (11)
100	4K Bytes (12)
101	Reserved
110	Reserved
111	Reserved

Bit 6	The number of SDRAM component banks
0	Two banks
1	Four banks

Bit 7	DRAM bank enable
0	Disable
1	Enable
Bits 8-15	Reserved

These two 16-bit registers define the configuration of each DRAM' s bank. Offset 0x30 configures bank 0, and offset 0x34 configures bank 1.

DRAM Bank 0/1 Base Address Register (DRAMBA0/DRAMBA1)

Port address: 0x00000032/36 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Base Address [0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Base Address [8:9]		Reserved					

Bits 0-9 DRAM base address

These two 16-bit registers define the most significant ten bits of each DRAM banks' base (bottom) address. The DRAM base address, together with the DRAM size as defined in DRAMCFG0/1, constitutes the whole address range of each DRAM bank.

Bits 10-15 Reserved

DRAM Control Register (DRAMCTRL)

Port address: 0x00000038 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Banks	Type	Onboard	TESTREF	Precharge	MRS	Refresh	CKE
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
DISAF	MCLK Control Select			MCLK Data Select			

Bit 0 Number of banks in DIMM

Bit 1 DRAM type
1 SDRAM

Bit 2 SDRAM configuration
0 DIMM
1 Onboard

Bit 3 SDRAM reset end select
Reserved for simulation only.

Bit 4 SDRAM precharge set
Setting this bit will issue a precharge command to SDRAM.

Bit 5 SDRAM mode register set
Setting this bit will issue a load mode register command to SDRAM.

Bit 6 SDRAM refresh cycle enable
Setting this bit will issue a refresh command to SDRAM.

Bit 7 SDRAM clock enable
0 Disable
1 Enable

Bit 8 Swap out 0xA0000 ~ 0xFFFFF
When this bit is set to logic 1, address space 0xA0000 ~ 0xFFFFF will not be recognized as system DRAM space.

Bits 9-11 DRAM controller control signal clock skew adjustment

DRAMCTRL [9:11]	MCLK Control Delay
000	0
001	2
010	4
011	6
100	8
101	10
110	12
111	14

Bits 12-14 DRAM controller data latch clock skew adjustment

DRAMCTRL [12:14]	MCLK Data Delay
000	0
001	2
010	4
011	6
100	8
101	10
110	12
111	14

DRAM Timing Control Register 0 (DRAMTCTL0)

Port address: 0x0000003a **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Refresh Rate [0:2]			CAS Latency [0:1]		ACMDdly[0:1]		RASact[0]
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
RASact[1:2]		RAS Precharge Time [0:2]			DRAM Cycle Time [0:2]		

Bits 0-2 Refresh rate

DRAMTCTL0 [0:2]	Refresh Rate
000	15 µs (default)
001	30 µs
010	60 µs
011	120 µs
100	240 µs
101	480 µs
110	960 µs
111	Disable

Bits 3-4 CAS latency

DRAMTCTL0 [3:4]	CAS Latency (MCLK)
00	Reserved
01	1
10	2
11	3 (default)

Bits 5-6 Active command delay

DRAMTCTL0 [5:6]	Active Command Delay (MCLK)
00	0
01	1
10	2
11	3 (default)

Bits 7-9 RAS# active pulse width t_{RAS}

DRAMTCTL0 [7:9]	RAS# Active Pulse Width (MCLK)
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8 (default)

Bits 10-12 RAS# precharge time t_{RP}

DRAMTCTL0 [10:12]	RAS# Precharge Time (MCLK)
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8 (default)

Bits 13-15 DRAM cycle time t_{RC}

DRAMTCTL0 [13:15]	DRAM Cycle Time (MCLK)
000	2
001	3
010	4
011	5
100	6
101	7
110	8

111	9 (default)

DRAM Timing Control Register 1 (DRAMTCTL1)

Port address: 0x0000003e **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved		Active to Read/Write Delay [0:2]			LDI2ACT[0:2]		

Bits 0-9 Reserved

Bits 10-12 Active to read or write delay t_{RCD}

DRAMTCTL1 [10:12]	**Active to Read/Write Delay (MCLK)**
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

Bits 13-15 Last data-in to active command period during write cycle

DRAMTCTL1 [13:15]	**Last Data-in to ACT Period (MCLK)**
000	1
001	2
010	3
011	4
100	5
101	6
110	7
111	8

DRAM Timing Control Register 2 (DRAMTCTL2)

Port address: 0x0000003e **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
LDI2PRE[0:1]		MCLK Freq. Select		ENECP	TEST	CLKDC[0:1]	
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
CLKDC[2]	MCLK_OUT_SEL [0:2]			PRE_ALL	Reserved		

Bits 0-1 Last data-in to precharge command during write cycle

DRAMTCTL2 [0:1]	CAS# Active Time (MCLK)
00	1
01	2
10	3
11	4

Bits 2-3 MCLK frequency select

DRAMTCTL2 [2:3]	MCLK Freq. Select
00	CPUCLK
01	CPUCLK/1.5
10	CPUCLK/2
11	Reserved

Bit 4 ECP enable

When this bit is set, the GPIO[0:12] and GPIO[15:18] are redefined as parallel port interface.

Bit 5 Test mode for outputting CPUCLK, MCLK_CTL, and MCLK_DATA

When this bit is set and bit-4 is reset, the above three internal clocks are showed on pins GPIO[10:12]. This mode is used to adjust phase skew of MCLK.

Bits 6-8 MCLK output buffer control

DRAMTCTL2 [6:8]	MCLK Driving Capacity (mA)
000	16

001	8
010	8
011	4
100	8
101	4
110	4
111	Tristate

Bits 9-11 MCLK output delay reference CLKTREE MCLKO

DRAMTCTL2 [9:11]	**MCLK Output Delay**
000	0
001	2
010	4
011	6
100	8
101	10
110	12
111	14

Bit 12 Precharge all enable

Setting this bit will allow SDRAM state machine always being through precharge state and stopping in idle state after each SDRAM read/write access.

Bit 13-15 Reserved

6.2 Video accelerator Registers

Video accelerator (VA) consists of video pre-engine (VPRE) and post-engine (VPOST). The VPRE handles the functions relating to video inputs from, for example MPEG chips or camera. The VPOST handles the functions relating to pictures displayed to TV (interlace) or monitor (non-interlace). There are totally thirty-three registers used to set up all the functions of VA. Among them, twenty-eight registers are in VPOST with I/O space ranging from 0xf0000100 to 0xf000019c, while the remaining five registers are in VPRE with I/O space ranging from 0xf00001c0 to 0xf00001dc.

6.2.1 VPOST Registers

The following table lists the twenty-eight registers that are included in the VPOST. The symbol BA found in the table stands for I/O base address 0xf0000000.

Port Address	Symbol	Access	Description
BA + 0x100	VPC	R/W	VPOST control register
BA + 0x104	OPWFSR	R/W	Background stream fetch stop/restart for opaque
BA + 0x108	WKSEX	R/W	Window key start/end X register
BA + 0x10c	WKSEY	R/W	Window key start/end Y register
BA + 0x110	HWCSWX	R/W	Hardware cursor start/width X register
BA + 0x114	HWCSHY	R/W	Hardware cursor start/height Y register
BA + 0x118	GFXSCKM	R/W	Graphics stream color key mask register
BA+ 0x11c	GFXSCK	R/W	Graphics stream color key register
BA + 0x120	OVLC	R/W	Overlay control register
BA + 0x124	GFXSSA	R/W	Graphics stream start address register
BA + 0x128	VASSA	R/W	VA stream start address register
BA + 0x12c	HWCSSA	R/W	Hardware cursor stream start address register
BA + 0x130	GFXVASS	R/W	Graphics/VA stream stride register
BA + 0x134	HWCSS	R/W	Hardware cursor stream stride register
BA + 0x138	GFXVASFF	R/W	Graphics/VA stream fetch finish register
BA + 0x13c	VASC	R/W	VA scaling control register
BA + 0x140	LUTINDEX	R/W	Look-up-table index register
BA + 0x144	LUTDATA	R/W	Look-up-table data register
BA + 0x148	FF12T	R/W	FIFO 1/2 threshold register
BA + 0x14c	FF34T	R/W	FIFO 3/4 threshold register
BA + 0x150	VAYCADJ	R/W	VA brightness/contrast/hue/saturation adjustment
BA + 0x154	SCF	R/W	Subcarrier frequency register
BA + 0x158	SCFIP	R/W	Subcarrier frequency initial phase register
BA + 0x15c	HTDEE	R/W	Horizontal total/display enable end register
BA + 0x160	HSYNCSE	R/W	HSYNC start/end register
BA + 0x164	VTDEE	R/W	Vertical total/display enable end register
BA + 0x168	VRSE	R/W	Vertical retrace start/end register

BA + 0x16c	HRS	R/W	Horizontal retrace start register
BA + 0x170	HWCBC	R/W	Hardware cursor background color register
BA + 0x174	HWCFC	R/W	Hardware cursor foreground color register
BA + 0x178	TVTWH	R/W	TV encoder test width/height register
BA + 0x17c	VPTC	R/W	VPOST test control register
BA + 0x180	FIFO1D	R/W	FIFO 1 data register
BA + 0x184	FIFO2D	R/W	FIFO 2 data register
BA + 0x188	FIFO3D	R/W	FIFO 3 data register
BA + 0x18c	FIFO4D	R/W	FIFO 4 data register
BA + 0x190	FIFO5D	R/W	FIFO 5 data register
BA + 0x194	DTORT	R	DTO ROM test register
BA + 0x198	VPTS	R	VPOST test status register
BA + 0x19c	VPCTD	R	VPOST counter test data register

VPOST Control Register (VPC)

Port address: 0x00000100 **Access type**: read/write **Default**: 0x00000000

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved				Wait_Yoff			
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved	CSW	HP	VP	Flicker Mode		BPP	
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
BGSEL	GFX_HUP	GFX_VUP	OFFSET	AJEN	AJCTL	FAL_D	DIG_ON
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
DISP_ON	FLK_ON	GFX_EN	VA_EN	OPA_EN	HWC_EN	WINKEN	COLKEN

Bits 0-3 Reserved

Bits 4-7 Wait_Yoff
Wait_Yoff = 2 x SCLK / 13.5MHz

Bit 8 Reserved

Bit 9 CbCr swap for 8-bit YCbCr video output mode
0 Output sequence is Cb0, Y0, Cr0, Y1, Cb2, Y2, Cr2, Y3, etc. (CCIR-656)
1 Output sequence is Cr0, Y0, Cb0, Y1, Cr2, Y2, Cb2, Y3, etc. (CCIR-656)

Bit 10 HSYNC output pin polarity
0 Negative sync pulse
1 Positive sync pulse

Bit 11 VSYNC output pin polarity
0 Negative sync pulse
1 Positive sync pulse

Bits 12- 13 3-line flicker-free filter mode
00 2D filter with 121, 242, 121 weightings
01 1D filter with 121 weightings
10 2D filter with 131, 141, 131 weightings
11 1D filter with 565 weightings

Bits 14-15 BPP (bits per pixel) for graphics stream

00	16-color mode
01	256-color mode
10	565 high color mode
11	Reserved
Bit 16	Background Select
0	Graphics
1	VA video
Bit 17	Graphics stream horizontal 2x up-scaling (replication)
0	Disable
1	Enable
Bit 18	Graphics stream vertical 2x up-scaling (replication)
0	Disable
1	Enable
Bit 19	Odd/even field data offset
0	One line offset
1	No offset
Bit 20	Brightness/contrast/hue/saturation adjustment enable
0	Disable
1	Enable
Bit 21	Brightness/contrast/hue/saturation adjustment control
0	Full-screen adjustment by using VACADJ and VAYADJ registers
1	Adjust data within windows defined by window keys only
Bit 22	P[0:7] Output Control
0	Pixel data is sampled by rising-edge of PCLK then output to P[0:7]
1	Pixel data is sampled by falling-edge of PCLK then output to P[0:7]
Bit 23	Digital video output enable
0	Disable
1	Enable
Bit 24	Screen on
0	Screen off
1	Screen on

Bit 25	3-line flicker-free filter enable
0	Disable
1	Enable
Bit 26	Graphics stream enable
0	Disable
1	Enable
Bit 27	VA stream enable
0	Disable
1	Enable
Bit 28	Window opaque enable
0	Disable
1	Enable
Bit 29	Hardware cursor enable
0	Disable
1	Enable
Bit 30	Window key enable
0	Disable
1	Enable
Bit 31	Color key enable
0	Disable
1	Enable

Background Stream Fetch Stop/Restart for Opaque Window Register (OPWFSR)

Port address: 0x00000104 **Access type**: read/write **Default**: 0x----

0	1	2	3	4	5	6	7
Reserved						OPWFR[0:1]	
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
OPWFR[2:9]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved						OPWFS[0:1]	
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
OPWFS[2:9]							

Bits 0-5 Reserved

Bits 6-15 Background stream fetch restart for opaque window

A 10-bit value specifying the horizontal offset in DWORD memory cycles the background stream is to restart fetching

Bits 16-21 Reserved

Bits 22-31 Background stream fetch stop for opaque window

A 10-bit value specifying the horizontal offset in DWORD memory cycles the background stream is to be hidden for opaque window display

Window Key Start/End X Register (WKSEX)

Port address: 0x00000108 **Access type**: read/write **Default**: 0x----

0	1	2	3	4	5	6	7
Reserved					WinKey_XS[0:2]		
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
WinKey_XS[3:10]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					WinKey_XE[0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
WinKey_XE[3:10]							

Bits 0-4 Reserved

Bits 5-15 Window key start X

An 11-bit value specifies the first horizontal pixel position of the window

Bits 16-20 Reserved

Bits 21-31 Window key end X

An 11-bit value specifies the last horizontal pixel position of the window

Window Key Start/End Y Register (WKSEY)

Port address: 0x0000010c **Access type**: read/write **Default**: 0x----

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">WinKey_YS[0:1]</td></tr>
<tr><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="8">WinKey_YS[2:9]</td></tr>
<tr><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">WinKey_YE[0:1]</td></tr>
<tr><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><td colspan="8">WinKey_YE[2:9]</td></tr>
</table>

Bits 0-5	Reserved
Bits 6-15	Window key start Y A 10-bit value specifies the first vertical scan line of the window
Bits 16-21	Reserved
Bits 22-31	Window key end Y A 10-bit value specifies the last vertical scan line of the window

Hardware Cursor Start/Width X Register (HWCSWX)

Port address: 0x00000110 **Access type**: read/write **Default**: 0x----

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="5">Reserved</td><td colspan="3">Hardware Cursor Start X [0:2]</td></tr>
<tr><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="8">Hardware Cursor Start X [3:10]</td></tr>
<tr><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><td>Reserved</td><td colspan="7">Hardware Cursor Width X [0:6]</td></tr>
</table>

Bits 0-4 Reserved

Bits 5-15 Hardware cursor start X

An 11-bit value specifies the horizontal starting pixel position of the hardware cursor

Bits 16-24 Reserved

Bits 25-31 Hardware cursor width X

A 7-bit value specifies the width of the hardware cursor (maximum allowable value = 64)

Hardware Cursor Start/Height Y Register (HWCSHY)

Port address: 0x00000114 **Access type**: read/write **Default**: 0x----

0	1	2	3	4	5	6	7
						H/W Cursor Start Y [0:1]	
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Hardware Cursor Start Y [2:9]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved	Hardware Cursor Height Y [0:6]						

Bits 0-5 Reserved

Bits 6-15 Hardware cursor start Y

A 10-bit value specifies the vertical starting scan line of the hardware cursor

Bits 16-24 Reserved

Bits 25-31 Hardware cursor height Y

A 7-bit value specifies the height of the hardware cursor (maximum allowable value = 64)

Graphics Stream Color Key Mask Register (GFXCKM)

Port address: 0x00000118 **Access type**: read/write **Default**: 0x----

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved						Color Key Mask [0:1]	
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Color Key Mask [2:9]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Color Key Mask [10:17]							

Bits 0-13	Reserved
Bits 14-19	Graphics stream color key mask blue
Bits 20-25	Graphics stream color key mask green
Bits 26-31	Graphics stream color key mask red

Graphics Stream Color Key Register (GFXCK)

Port address: 0x0000011c **Access type**: read/write **Default**: 0x----

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved						Color Key [0:1]	
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Color Key [2:9]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Color Key [10:17]							

Bits 0-13	Reserved
Bits 14-19	Graphics stream color key blue
Bits 20-25	Graphics stream color key green
Bits 26-31	Graphics stream color key red

Overlay Control Register (OVLC)

Port address: 0x00000120 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved				OC0	OC1	OC2	OC3

Bits 0-27 Reserved

Bits 28-31 Overlay display select

These four 1-bit registers are selected by color key and window key as described below:

Color Key	Window Key	Overlay Control
0	0	OC0
0	1	OC1
1	0	OC2
1	1	OC3

Each 1-bit register, when selected, is used to control current overlaying output as described below.

Color Key	Overlay Output
0	Graphics stream
1	VA stream

Notes: Background on the screen, controlled by OC0 when all keys are inactive, should be either graphics stream or VA stream as specified by VPOSTCR_16. It means that OC0 should be programmed to either 0 (when VPOSTCR_16 = 1) or 1 (when VPOSTCR_16 = 0).

Graphics Stream Start Address Register (GFXSSA)

Port address: 0x00000124 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved		GFXSA[0:5]					
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
GFXSA[6:13]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
GFXSA[14:21]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
GFXSA[22:29]							

Bits 0-1 Reserved

Bits 2-31 Graphics stream start address

A 30-bit value specifies the offset in DWORD boundary from the beginning of the frame buffer for graphics data stream.

Hardware Cursor Stream Start Address Register (HWCSSA)

Port address: 0x0000012c **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved		HWCSA[0:5]					
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
HWCSA[6:13]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
HWCSA[14:21]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
HWCSA[22:29]							

Bits 0-1 Reserved

Bits 2-31 Hardware cursor stream start address

A 30-bit value specifies the offset in DWORD boundary from the beginning of the frame buffer for hardware cursor data stream.

Graphics/VA Stream Stride Register (GFXVASS)

Port address: 0x00000130 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved					GFXSS[0:2]		
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
GFXSS[3:10]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					VASS[0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
VASS[3:10]							

Bits 0-4 Reserved

Bits 5-15 Graphics stream stride

This register specifies the DWORD offset of vertically adjacent pixels in the graphics stream.

Bits 16-20 Reserved

Bits 21-31 VA stream stride

The register specifies the DWORD offset of vertically adjacent pixel in the VA stream buffer.

Hardware Cursor Stream Stride Register (HWCSS)

Port address: 0x00000134 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					HWCSS[0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
HWCSS[3:10]							

Bits 0-20 Reserved

Bits 21-31 Hardware cursor stream stride

This register specifies the DWORD offset of vertically adjacent pixels in the hardware cursor stream buffer.

Graphic/VA Stream Fetch Finish Register (GFXVASFF)

Port address: 0x00000138 **Access type**: read/write **Default**: 0x---

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">GFXSFF[0:1]</td></tr>
<tr><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
<tr><td colspan="8">GFXSFF[2:9]</td></tr>
<tr><td>16</td><td>17</td><td>18</td><td>19</td><td>20</td><td>21</td><td>22</td><td>23</td></tr>
<tr><td colspan="6">Reserved</td><td colspan="2">VASFF[0:1]</td></tr>
<tr><td>24</td><td>25</td><td>26</td><td>27</td><td>28</td><td>29</td><td>30</td><td>31</td></tr>
<tr><td colspan="8">VASFF[2:9]</td></tr>
</table>

Bits 0-5 Reserved

Bits 6-15 Graphics stream fetch finish

This register specifies the number of DWORD DRAM access cycles for a horizontal scan line fetching of graphics data stream.

Bits 16-21 Reserved

Bits 22-31 VA stream fetch finish

This register specifies the number of DWORD DRAM access cycles for a horizontal scan line fetching of VA video data stream.

VA Scaling Control Register (VASC)

Port address: 0x0000013c **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
VUPS	Reserved		VA Vertical Scaling Factor [0:4]				
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
VA Vertical Scaling Factor [5:12]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
HUPS	Reserved		VA Horizontal Scaling Factor [0:4]				
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
VA Horizontal Scaling Factor [5:12]							

Bit 0 VA vertical up-scaling method

0 Replication

1 Interpolation

Bits 1-2 Reserved

Bits 3-15 VA vertical scaling factor

This 13-bit value specifies the vertical scaling factor of 0.5 (1/2 down-scaling), and 1.0 ~ 7.999 (up-scaling). Bits 0-2 specify the integral part and bits 3-12 specify the decimal part of the scaling factor. The 1/2 downing-scaling will be done when this value < 1 (bits 0-2 = 000 and bits 3-12 don' t care). Scaling is disabled when this value = 1.000 (bits 0-2 = 001 and bits 3-12 = 000H).

Bit 16 VA horizontal up-scaling method

0 Replication

1 Interpolation

Bits 17-18 Reserved

Bits 19-31 VA horizontal scaling factor

This 13-bit value specifies the horizontal scaling factor of 0.5 (1/2 down-scaling), and 1.0 ~ 7.999 (up-scaling). Bits 0-2 specify the integral part and bits 3-12 specify the decimal part of the scaling factor. The 1/2 downing-scaling will be done when this value < 1 (bits 0-2 = 000 and bits 3-12 don' t care). Scaling is disabled when this value = 1.000 (bits 0-2 = 001 and bits 3-12 = 000H).

Lookup Table Index Register (LUTINDEX)

Port address: 0x00000140 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved							R/W
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
LUT Index [0:7]							

Bits 0-22 Reserved

Bit 23 Lookup table (LUT) read/write mode

Bits 24-31 Lookup table index

This index determines which color lookup table location will be accessed. The color lookup table is used for color mapping between pixel value of graphics stream in pseudo modes (16- and 256-color modes) and the display color on the screen.

Lookup Table Data Register (LUTDATA)

Port address: 0x00000144 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved		LUT Red Data [5:0]					
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved		LUT Green Data [5:0]					
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved		LUT Blue Data [5:0]					

Bits 0-9	Reserved
Bits 10-15	Specify the lookup table red data
Bits 16-17	Reserved
Bits 18-23	Specify the lookup table green data
Bits 24-25	Reserved
Bits 26-31	Specify the lookup table blue data

FIFO 1/2 Threshold Register (FF12T)

Port address: 0x00000144 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved			SRM1_HT				
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved			SRM1_LT				
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved			SRM2_HT				
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved			SRM2_LT				

Bits 0-2 Reserved

Bits 3-7 First FIFO high threshold

When the first frame buffer FIFO is filled up to this threshold, the FIFO is ready to release DRAM access to other pending requests. The initial value is 18H.

Bits 8-10 Reserved

Bits 11-15 First FIFO low threshold

When the first frame buffer FIFO is fetched down to this threshold by graphics controller, a request is generated to the DRAM controller for DRAM access. The initial value is 0CH.

Bits 16-18 Reserved

Bits 19-23 Second FIFO high threshold

When the second frame buffer FIFO is filled up to this threshold, the FIFO is ready to release DRAM access to other pending requests. The initial value is 18H.

Bits 24-26 Reserved

Bits 27-31 Second FIFO low threshold

When the second frame buffer FIFO is fetched down to this threshold by graphics controller, a request is generated to the DRAM controller for DRAM access. The initial value is 0CH.

FIFO 3/4 Threshold Register (FF34T)

Port address: 0x0000014c **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved			SRM3_HT				
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved			SRM3_LT				
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved			SRM4_HT				
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved			SRM4_LT				

Bits 0-2 Reserved

Bits 3-7 Third FIFO high threshold

When the third frame buffer FIFO is filled up to this threshold, the FIFO is ready to release DRAM access to other pending requests. The initial value is 18H.

Bits 8-10 Reserved

Bits 11-15 Third FIFO low threshold

When the third frame buffer FIFO is fetched down to this threshold by graphics controller, a request is generated to the DRAM controller for DRAM access. The initial value is 0CH.

Bits 16-18 Reserved

Bits 19-23 Fourth FIFO high threshold

When the fourth frame buffer FIFO is filled up to this threshold, the FIFO is ready to release DRAM access to other pending requests. The initial value is 18H.

Bits 24-26 Reserved

Bits 27-31 Fourth FIFO low threshold

When the fourth frame buffer FIFO is fetched down to this threshold by graphics controller, a request is generated to the DRAM controller for DRAM access. The initial value is 0CH.

VA Stream Brightness/Contrast/Hue/Saturation Adj. Register (VAYCADJ)

Port address: 0x00000150 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved				Contrast [0:3]			
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Brightness [0:7]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Saturation [0:3]				Reserved	Hue [0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Hue [3:10]							

Bits 0-3 Reserved

Bits 4-7 VA contrast value

A 4-bit contrast adjustment value allows adjustments in contrast from 1/8 to 15/8, in increments of 1/8. Bit 4 specifies the integral part and bits 5-7 specify the decimal part of this value. Contrast adjustment is implemented by multiplying the Y data by this constant.

Bits 8-15 VA brightness value

An 8-bit 2' -complement value allows adjustments in brightness from -128 to +127, in increments of 1. The brightness adjustment is implemented by adding or subtracting this constant to/from the Y data.

Bits 16-19 VA saturation value

A 4-bit saturation adjustment value allows adjustments in saturation from 1/8 to 15/8, in increments of 1/8. Bit 16 specifies the integral part and bits 17-19 specify the decimal part of this value. Saturation adjustment is implemented by multiplying both Cb and Cr by this constant.

Bit 20 Reserved

Bits 21-31 VA hue value

An 11-bit hue adjustment value allows adjustments in hue from 0 degree to 360 degree, in increments of 0.176 degree. Hue adjustment is implemented by

Cb1 = Cb cosA + Cr sinA

Cr1 = Cr cosA - Cb sinA

Subcarrier Frequency Register (SCF)

Port address: 0x00000154 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
SCF[0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
SCF[8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
SCF[16:23]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
SCF[24:31]							

Bits 0-31

Subcarrier frequency

A 32-bit value specifies the subcarrier frequency for TV by using the following equation:

SCF value = (fsc/fvoclk) 2^32

Subcarrier frequency is generated from the stable VOCLK (27 MHz) by an internal DDA (Digital Differential Accumulator).

Subcarrier Frequency Initial Phase Register (SCFIP)

Port address: 0x00000158 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Subcarrier Frequency Initial Phase [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Subcarrier Frequency Initial Phase [8:15]							

Bits 0-15 Reserved

Bits 16-31 Subcarrier frequency initial phase

This 16-bit register specifies the initial phase between the color subcarrier and sync signal.

Horizontal Total/Display Enable End Register (HTDEE)

Port address: 0x0000015c **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved					Horizontal Total [0:2]		
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Horizontal Total [3:10]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					Horizontal Display End [0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Horizontal Display End [3:10]							

Bits 0-4 Reserved

Bits 5-15 Horizontal total

This 11-bit value specifies the total number of pixels in the horizontal scan line interval including the retrace time.

Bits 16-20 Reserved

Bits 21-31 Horizontal display enable end

This 11-bit value specifies the total number of displayed pixels for one scan line.

HSYNC Start/End Register (HSYNCSE)

Port address: 0x00000160 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved					HSYNC Start [0:2]		
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
HSYNC Start [3:10]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					HSYNC End [0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
HSYNC End [3:10]							

Bits 0-4 Reserved

Bits 5-15 HSYNC start

An 11-bit value, programmed in pixels, at which the HSYNC signal becomes active.

Bits 16-20 Reserved

Bits 21-31 HSYNC end

An 11-bit value, programmed in pixels, at which the HSYNC signal becomes inactive.

Vertical Total/Display Enable Register (VTDEE)

Port address: 0x00000164 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved						Vertical Total [0:1]	
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Vertical Total [2:9]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved						VDEE [0:1]	
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
VDEE [2:9]							

Bits 0-5 Reserved

Bits 6-15 Vertical total

This 10-bit value specifies the total number of scan lines for one field on the screen, including the retrace time.

Bits 16-21 Reserved

Bits 22-31 Vertical display enable end

This 10-bit value specifies the total number of scan lines displayed for one field on the screen.

Vertical Retrace Start/End Register (VRSE)

Port address: 0x00000168 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved						VRS [0:1]	
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
VRS [2:9]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved						VRE [0:1]	
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
VRE [2:9]							

Bits 0-5 Reserved

Bits 6-15 Vertical retrace start

This 10-bit value specifies the scan line at which the vertical retrace becomes active.

Bits 16-21 Reserved

Bits 22-31 Vertical retrace end

This 10-bit value specifies the scan line at which the vertical retrace becomes inactive.

Horizontal Retrace Start Register (HRS)

Port address: 0x0000016c **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					Horizontal Retrace Start [0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Horizontal Retrace Start [3:10]							

Bits 0-20 Reserved

Bits 21-31 Horizontal retrace start

This 11-bit value specifies the pixel at which the internal horizontal retrace becomes active. The internal horizontal retrace pulse width is fixed to 16 pixel clock cycles.

Hardware Cursor Background Color Register (HWCBC)

Port address: 0x00000170 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Hardware Cursor Background Color [0:7]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Hardware Cursor Background Color [8:15]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Hardware Cursor Background Color [16:23]							

Bits 0-7 Reserved

Bits 8-31 Hardware cursor background color

This 24-bit value specifies the background color for the hardware cursor. Among them, bits 8-15 have the red value, bits 16-23 the green value, and bits 24-31 the blue value. Only the color mode in RGB 8:8:8 is allowed.

Hardware Cursor Foreground Color Register (HWCFC)

Port address: 0x00000174 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Hardware Cursor Foreground Color [0:7]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Hardware Cursor Foreground Color [8:15]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Hardware Cursor Foreground Color [16:23]							

Bits 0-7 Reserved

Bits 8-31 Hardware cursor foreground color

This 24-bit value specifies the foreground color for the hardware cursor. Among them, bits 8-15 have the red value, bits 16-23 the green value, and bits 24-31 the blue value. Only the color mode in RGB 8:8:8 is allowed.

TV Encoder Test Width/Height Register (TVTWH)

Port address: 0x00000178 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
				TEST_C	TEST_CP	TEST_Y	TESTDAC
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
VREFSEL	CPOFF			Analog Video Output		TV System	
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
TV Encoder Horizontal Test Width							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
TV Encoder Vertical Test Height							

Bits 0-3 Reserved

Bits 4-7 Digital output selection for test

Bit 7 - enable bit; bit 6 - select TV luminance Y; bit 5 - select TV composite CP; bit 4 - select TV chrominance C

Bit 8 DAC reference voltage select

Bits 9-11 DAC enable

Bits 12-13 Analog video output mode

0x - RGB out, TV-encoder is off; 10 - composite video; 11 - S-video + composite video

Bits 14-15 TV system

00 - PAL-B, D, G, H, N; 01 - PAL_M; 10 –NTSC; 11 –Reserved

Bits 16-23 TV encoder horizontal test width

This 8-bit value specifies the total horizontal TV scan lines when the TV encoder horizontal test is enabled (VPTC[29] = 1). This register is not used during normal operation.

Bits 24-31 TV encoder vertical test height

This 8-bit value specifies the total vertical TV scab lines when the TV encoder vertical test is enabled (VPTC[28] = 1). This register is not used during normal operation.

VPOST Test Control Register (VPTC)

Port address: 0x0000017c **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved			ARM	PRESET	SELF	Signature Input Select	
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved	VTEST	Counter Select					
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
State Machine Select			Test Mode	TV_V	TV_H	Color Bar	Gray Level

Bits 0-10 Reserved

Bit 11 Arm signature analyzer
0 Disable
1 Start signature analyzer operation

Bit 12 Preset signature to seed value
0 Preset signature to seed value when analysis begins
1 Do not preset

Bit 13 Signature analyzer self test
0 Disable
1 Enable

Bits 14-15 Signature analyzer input select

Bits 14-15	Signature Analyzer Input Data
00	Null (all zeroes)
01	CP/R data
10	Y/G data
11	C/B data

Bit 16 Reserved

Bit 17 Digital video pin I/O mode
0 Video in
1 Video output

Bits 18-23	VPOST counter test select
000000	SRM1 x address counter
000001	SRM2 x address counter
000010	SRM3 x address counter
000011	SRM4 x address counter
000100	SRM5 x address counter
000101	SRM1 y address counter
000110	SRM2 y address counter
000111	SRM3 y address counter
001000	SRM4 y address counter
001001	SRM5 y address counter
001010	SRM1 and SRM2 opaque counter in MCLK
001011	SRM3, SRM4, and SRM5 opaque counter in MCLK
001100	VA scaling counter
001101	CRTC horizontal and vertical counter
001110	TV horizontal and vertical counter
001111	Opaque counter in VOCLK

Bits 24-26	SRM state machine select
000	State machine of SRM1
001	State machine of SRM2
010	State machine of SRM3
011	State machine of SRM4
100	State machine of SRM5

Bit 27	VPOST test enable
0	Disable
1	Enable

Bit 28	TV encoder vertical test enable
0	Disable
1	Enable

Bit 29	TV encoder horizontal test enable
0	Disable
1	Enable

Bit 30	Color bar test enable
0	Disable
1	Enable

Bit 31	Gray level test enable
0	Disable
1	Enable

FIFO 1 Data Register (FIFO1D)

Port address: 0x00000180 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
FIFO1D[0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
FIFO1D[8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
FIFO1D[16:23]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
FIFO1D[24:31]							

Bits 0-31 FIFO 1 data

A read or write access to the FIFO1D register will incremet the FIFO address. The FIFO address will be reset to 0 by a write access to the VPOSTCR_27 register and setting it to 1.

FIFO 2 Data Register (FIFO2D)

Port address: 0x00000184 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
FIFO2D[0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
FIFO2D[8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
FIFO2D[16:23]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
FIFO2D[24:31]							

Bits 0-31 FIFO 2 data

A read or write access to the FIFO2D register will incremet the FIFO address automatically. The FIFO address will be reset to 0 by a write access to the VPOSTCR_27 register and setting it to 1.

FIFO 3 Data Register (FIFO3D)

Port address: 0x00000188 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
FIFO3D[0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
FIFO3D[8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
FIFO3D[16:23]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
FIFO3D[24:31]							

Bits 0-31 FIFO 3 data

A read or write access to the FIFO3D register will incremet the FIFO address automatically. The FIFO address will be reset to 0 by a write access to the VPOSTCR_27 register and setting it to 1.

FIFO 4 Data Register (FIFO4D)

Port address: 0x0000018c **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
FIFO4D[0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
FIFO4D[8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
FIFO4D[16:23]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
FIFO4D[24:31]							

Bits 0-31 FIFO 4 data

A read or write access to the FIFO4D register will incremet the FIFO address automatically. The FIFO address will be reset to 0 by a write access to the VPOSTCR_27 register and setting it to 1.

FIFO 5 Data Register (FIFO5D)

Port address: 0x00000190 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
FIFO5D[0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
FIFO5D[8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
FIFO5D[16:23]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
FIFO5D[24:31]							

Bits 0-31 FIFO 5 data

A read or write access to the FIFO5D register will incremet the FIFO address automatically. The FIFO address will be reset to 0 by a write access to the VPOSTCR_27 register and setting it to 1.

DTO ROM Test Register (DTORTR)

Port address: 0x00000194 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
DTO ROM [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
DTO ROM [8:15]							

Bits 0-15 Reserved

Bits 16-31 DTO ROM test

A read or write access to the DTORTR register will increment the FIFO address automatically. The FIFO address will be reset to 0 by a write access to the VPOSTCR_27 register and setting it to 1.

VPOST Test Status Register (VPTS)

Port address: 0x00000198 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved			VPOST State Machine Status				
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved						DEN	CRC-Busy
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Signature Data [0:7]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Signature Data [8:15]							

Bits 0-2	Reserved
Bits 3-7	VPOST state machine status
Bits 8-13	Reserved
Bit 14	Vertical display enable status
0	Inactive
1	Active
Bit 15	Signature analyzer status
0	Idle
1	Busy
Bits 16-31	Signature data

VPOST Counter Test Data Register (VPCTD)

Port address: 0x0000019c **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved		VPOST Counter Test Data [0:5]					
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
VPOST Counter Test Data [6:13]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
VPOST Counter Test Data [14:21]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
VPOST Counter Test Data [22:29]							

Bits 0-1	Reserved
Bits 2-31	VPOST counter test data This register contains data output of counter under testing when the VPOST counter test is enabled (VPTCR_27 = 1). The counter under testing is determined by VPOST counter test select register (VPTC[18-23]).

6.2.2 VPRE Registers

There are eight registers, with I/O space allocated from 0xf00001c0 to 0xf00001dc, included in VPRE.

Port Address	Symbol	Access	Description
BA + 0x1c0	VCC	R/W	Video Capture Control Register
BA + 0x1c4	CWSEX	R/W	Cropping Window Start/End X Register
BA + 0x1c8	CWSEY	R/W	Cropping Window Start/End Y Register
BA + 0x1cc	CVHW	R/W	Captured Video Height/Width Register
BA + 0x1d0	CSA0	R/W	Capture Frame Buffer 0 Start Address Register
BA + 0x1d4	CSA1	R/W	Capture Frame Buffer 1 Start Address Register
BA + 0x1d8	CTM	R/W	Capture Test Mode Register
BA + 0x1dc	CSTMD	R	Capture SRAM Test Mode Data Register
BA + 0x1e0	VIM	R/W	Video Interrupt Mode

Video Capture Control Register (VCC)

Port address: 0x000001c0 **Access type**: read/write **Default**: 0x180c0000

0	1	2	3	4	5	6	7
Reserved			VCAP_HT[4:0]				
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved			VCAP_LT[4:0]				
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					RES_ON	PWOFF	CKF
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
DBE	DBS	Byte Swap		HSP	VSP	SKP	VCEN

Bits 0-2 Reserved

Bits 3-7 Video capture FIFO high threshold

When the video capture FIFO is filled to this threshold, a request is sent to the DRAM controller for DRAM access. The initial value is 18H.

Bits 8-10 Reserved

Bits 11-15 Video capture FIFO low threshold

When the video capture FIFO is fetched to this threshold by DRAM controller, the FIFO is ready to release DRAM access to other pending requests. The initial value is 0cH.

Bits 16-20 Reserved

Bit 21 Interlace mode, odd and even field display

0 Turn off

1 Turn on

Bit 22 Video in, power down off

0 Turn off

1 Turn on

Bit 23 VCLK falling edge latch

0 Input video data and signals are latched by rising edge of VCLK

1 Input video data and signals are latched by falling edge of VCLK

Bit 24	Double buffering enable
0	Buffer 0 active
1	Buffer 1 active

Bit 25	Double buffering status (read only)
0	Buffer 0 active
1	Buffer 1 active

Bits 26-27 Input video stream format

Bits 26-27	YUV Input Video Stream Format 8-bit Mode
00	Y, U, Y, V, …
01	U, Y, V, Y, …
10	Y, V, Y, U, …
11	V, Y, U, Y, …

Bit 28	HS input pin polarity
0	Negative sync pulse
1	Positive sync pulse

Bit 29	VS input pin polarity
0	Negative sync pulse
1	Positive sync pulse

Bit 30	Skip field (interlaced) or frame (non-interlaced)
0	Capture all received field/frame video data
1	Capture every other received field/frame video data

Bit 31	Video capture enable
0	Disable
1	Enable

Cropping Window Start/End X Register (CWSEX)

Port address: 0x000001c4 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved				Cropping Window Start X [11:8]			
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Cropping Window Start X [7:0]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved				Cropping Window End X [11:8]			
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Cropping Window End X [7:0]							

Bits 0-3 Reserved

Bits 4-15 Cropping window start X

A 12-bit value specifies the number of pixels between the inactive edge of HS and the first cropped video pixel.

Bits 16-19 Reserved

Bits 20-31 Cropping window end X

A 12-bit value specifies the number of pixels between the inactive edge of HS and the last cropped video pixel.

Cropping Window Start/End Y Register (CWSEY)

Port address: 0x000001c8 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved					Cropping Window Start Y [10:8]		
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Cropping Window Start Y [7:0]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					Cropping Window End Y [10:8]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Cropping Window End Y [7:0]							

Bits 0-4 Reserved

Bits 5-15 Cropping window start Y

An 11-bit value specifies the number of pixels between the inactive edge of VS and the first cropped video pixel.

Bits 16-20 Reserved

Bits 21-31 Cropping window end Y

An 11-bit value specifies the number of pixels between the inactive edge of VS and the last cropped video pixel.

Captured Video Height/Width Register (CVHW)

Port address: 0x000001cc **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved					Capture Video Height [10:8]		
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Capture Video Height [7:0]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					Capture Video Width [10:8]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Capture Video Width [7:0]							

Bits 0-4 Reserved

Bits 5-15 Captured video height

An 11-bit value specifies the height in line of the captured video that is downscaled (or not) from the cropped video. Downscaling is automatically done by an internal DDA (Digital Differential Accumulator)

Bits 16-20 Reserved

Bits 21-31 Captured video width

An 11-bit value specifies the width in pixel of the captured video that is downscaled (or not) from the cropped video. Downscaling is automatically done by an internal DDA (Digital Differential Accumulator)

Capture Frame Buffer 0 Start Address Register (CSA0)

Port address: 0x000001d0 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved		Buffer 0 Start Address [29:24]					
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Buffer 0 Start Address [23:16]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Buffer 0 Start Address [15:8]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Buffer 0 Start Address [7:0]							

Bits 0-1 Reserved

Bits 2-31 Capture frame buffer 0 start address

A 30-bit value specifies the offset in DWORD boundary from the start of the captured video frame buffer 0.

Capture Frame Buffer 1 Start Address Register (CSA1)

Port address: 0x000001d4 **Access type**: read/write **Default**: 0x---

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="2">Reserved</td><td colspan="6">Buffer 1 Start Address [29:24]</td></tr>
<tr><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="8">Buffer 1 Start Address [23:16]</td></tr>
<tr><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th></tr>
<tr><td colspan="8">Buffer 1 Start Address [15:8]</td></tr>
<tr><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><td colspan="8">Buffer 1 Start Address [7:0]</td></tr>
</table>

Bits 0-1 Reserved

Bits 2-31 Capture frame buffer 1 start address

A 30-bit value specifies the offset in DWORD boundary from the start of the captured video frame buffer 1.

Capture Test Mode Register (CTM)

Port address: 0x000001d8 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Status of Power-on Setting [0:3]				Reserved			
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved			TME	Counter Test Mode Select [0:3]			
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Counter Test Mode Data [15:8]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Counter Test Mode Data [7:0]							

Bits 0-3 Status of power-on setting

These four bits memorize states of **MD[20~23]** during power-on interval. Firmware reads these bits to know how target board is configured.

Bits 4-10 Reserved

Bit 11 Capture test mode enable

0	Disable
1	Enable

Bits 12-15 Capture counter test mode select

0001	Cropping horizontal-counter
0010	Cropping vertical-counter
0100	Horizontal downscaling DDA
1000	Vertical downscaling DDA

Bits 16-31 Capture counter test mode data (read only)

Capture SRAM Test Mode Data Register (CSTMD)

Port address: 0x000001dc **Access type**: read/write **Default**: 0x---

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td colspan="8">Capture SRAM Test Mode Data [31:24]</td></tr>
<tr><th>8</th><th>9</th><th>10</th><th>11</th><th>12</th><th>13</th><th>14</th><th>15</th></tr>
<tr><td colspan="8">Capture SRAM Test Mode Data [23:16]</td></tr>
<tr><th>16</th><th>17</th><th>18</th><th>19</th><th>20</th><th>21</th><th>22</th><th>23</th></tr>
<tr><td colspan="8">Capture SRAM Test Mode Data [15:8]</td></tr>
<tr><th>24</th><th>25</th><th>26</th><th>27</th><th>28</th><th>29</th><th>30</th><th>31</th></tr>
<tr><td colspan="8">Capture SRAM Test Mode Data [7:0]</td></tr>
</table>

Bits 0-31	Capture SRAM test mode data

Video INTR Mode (VIM)

Port address: 0x000001e0 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved				FLT_ON	VSI_ON	VSO_ON	CAP_ON
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
IODD	OODD			FLT_INT	VSI_INT	VSO_INT	CAP_INT

Bits 0-19 Reserved

Bit 20 VPRE filter complete interrupt mode
0 Turn off
1 Turn on

Bit 21 Video in VSYNC interrupt mode
0 Turn off
1 Turn on

Bit 22 Video out VSYNC interrupt mode
0 Turn off
1 Turn on

Bit 23 Capture complete interrupt mode
0 Turn off
1 Turn on

Bits 24-27 Reserved

Bit 28 VPRE filter complete interrupt occur

Bit 29 Video in VSYNC interrupt occurred

Bit 30 Video out VSYNC interrupt occurred

Bit 31 Capture complete interrupt occurred

6.3 DMA REGISTERS

There are twelve registers included in two-channel Direct Memory Access (DMA) controller. The I/O address map is allocated from 0xf0000200 to 0xf000022c.

Port Address	Symbol	Access	Description
BA + 0x200	SAR0	R/W	Channel 0 Source Address Register
BA + 0x204	TAR0	R/W	Channel 0 Target Address Register
BA + 0x208	LETH0	R/W	Channel 0 Length Register
BA + 0x20c	MOD0	R/W	Channel 0 Mode Control Register
BA + 0x210	SAR1	R/W	Channel 1 Source Address Register
BA + 0x214	TAR1	R/W	Channel 1 Target Address Register
BA + 0x218	LETH1	R/W	Channel 1 Length Register
BA + 0x21c	MOD1	R/W	Channel 1 Mode Control Register
BA + 0x220	DBA0	R/W	DMA I/O Device 0 Bass Address
BA + 0x224	DBA1	R/W	DMA I/O Device 1 Base Address
BA + 0x228	LCAR0	R	Channel 0 Length Counter
BA + 0x22c	LCAR1	R	Channel 1 Length Counter

Channel 0 Source Address Register (SAR0)

Port address: 0xf0000200 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Channel 0 Source Address Register Byte 0							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Channel 0 Source Address Register Byte 1							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Channel 0 Source Address Register Byte 2							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Channel 0 Source Address Register Byte 3							

Bits 0-31 Channel 0 source address register

Define DMA transfer source address. In memory-to-memory mode, the source address should be set in word boundary.

Channel 0 Target Address Register (TAR0)

Port address: 0xf0000204 **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Channel 0 Target Address Register Byte 0							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Channel 0 Target Address Register Byte 1							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Channel 0 Target Address Register Byte 2							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Channel 0 Target Address Register Byte 3							

Bits 0-31 Channel 0 target address register

Define DMA transfer target address. In memory-to-memory mode, the target address should be set in word boundary.

Channel 0 Length Register (LETH0)

Port address: 0xf0000208 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							LEN [0]
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
LEN[1:8]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
LEN[9:16]							

Bits 0-14 Reserved

Bits 15-31 Channel 0 transfer length

LEN0[0-16] indicate the length of DMA transfer with maximum 128K bytes. In memory-to-memory transfer mode, the length must in word boundary, because of counting by word in length counter.

Channel 0 Mode Control Register (MOD0)

Port address: 0xf000020c **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
DMAen	Reserved	TCIen	ECPen	TC	M2M	DEM	IOtype[0]
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
IOtype[1]	TRtype[0:1]		IOrec				
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Wstate					FIX	Reserved	Tout[0]
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Tout[1:8]							

Bit 0	DMA enable
Bit 1	Reserved
Bit 2	Terminal count interrupt enable
0	Disable
1	Enable (once this bit is set and TC is asserted, the DMAC will generate external interrupt to host)
Bit 3	Enable ECP as DMA device
0	Disable
1	Enable
Bit 4	Terminal count flag
	TC = 1 (asserted) indicates the length counter reaching 0
Bit 5	Memory-to-memory transfer mode
0	Disable
1	Enable
Bit 6	Demand mode or block mode select (this bit is valid only in the transfer between memory and I/O)
0	The DMA transfer between memory and I/O is in block mode
1	The DMA transfer between memory and I/O is in demand mode
Bits 7-8	DMA I/O device type (only 8-bit external I/O devices is supported)
00	8-bit type, length counter counts by byte
01	16-bit type, length counter counts by half-word

10	32-bit type, length counter counts by word
11	Undefined
Bits 9-10	DMA transfer type
00	Memory-to-memory transfer
01	Memory-to-I/O transfer
1X	I/O-to-memory transfer
Bits 11-15	DMA I/O read/write command recovery time
	This field defines the recovery cycles between two read/write commands.
Bits 16-20	DMA I/O read/write command wait state
	This field defines the I/O read/write command wait state.
Bit 21	DMA transfer fix mode
0	The DMA is set in fix mode. The channel 0 is the most privileged. The channel 1 will not get the service token unless the channel 0 releases the request.
1	The DMA is set in rotate mode. In rotate mode, the DMA controller acknowledges channel 1 request right after channel 0 being served. The channel 0 and channel 1 are served by turns.
Bit 22	Reserved
Bits 23-31	Ready timeout counter
1	When I/O read/write command is issued, and if the I/O device inserts wait state by asserting IORDY, the ready timeout counter starts to count. If the counter reaches the Tout[0:8] before read/write command is completed, the timeout flags TO0 or TO1 is to be set.

Channel 1 Source Address Register (SAR1)

Port address: 0xf0000210 **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Channel 1 Source Address Register Byte 0							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Channel 1 Source Address Register Byte 1							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Channel 1 Source Address Register Byte 2							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Channel 1 Source Address Register Byte 3							

Bits 0-31 Channel 1 source address register

Define DMA transfer source address. In memory-to-memory mode, the source address should be set in word boundary.

Channel 1 Target Address Register (TAR1)

Port address: 0xf0000214 **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Channel 1 Target Address Register Byte 0							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Channel 1 Target Address Register Byte 1							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Channel 1 Target Address Register Byte 2							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Channel 1 Target Address Register Byte 3							

Bits 0-31 Channel 1 target address register

Define DMA transfer target address. In memory-to-memory mode, the target address should be set in word boundary.

Channel 1 Length Register (LETH1)

Port address: 0xf0000218 **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							LEN[0]
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
LEN[1:8]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
LEN[9:16]							

Bits 0-14	Reserved
Bits 15-31	Channel 1 transfer length LEN[0-16] indicate the length of DMA transfer with maximum 128K bytes. In memory-to-memory transfer mode, the length must in word boundary, because of counting by word in length counter.

Channel 1 Mode Control Register (MOD1)

Port address: 0xf000020c **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
DMAen	Reserved	TCIen	ECPen	TC	M2M	DEM	IOtype[0]
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
IOtype[1]	TRtype[0:1]		IOrec				
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Wstate					DACK0L	DACK1L	CS0L
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
CS1L	DACK1A	TO0	TO1	Reserved			

Bit 0 DMA enable

Bit 1 Reserved

Bit 2 Terminal count interrupt enable

0 Disable

1 Enable (once this bit is set and TC is asserted, the DMAC will generate external interrupt to host)

Bit 3 Enable ECP as DMA device

0 Disable

1 Enable

Bit 4 Terminal count flag

TC = 1 (asserted) indicates the length counter reaching 0

Bit 5 Memory-to-memory transfer mode

0 Disable

1 Enable

Bit 6 Demand mode or block mode select (this bit is valid only in the transfer between memory and I/O)

0 The DMA transfer between memory and I/O is in block mode

1 The DMA transfer between memory and I/O is in demand mode

Bits 7-8 DMA I/O device type (only 8-bit external I/O devices is supported)

00 8-bit type, length counter counts by byte

01 16-bit type, length counter counts by half-word

10 32-bit type, length counter counts by word

11 Undefined

Bits 9-10 DMA transfer type

00 Memory-to-memory transfer

01 Memory-to-I/O transfer

1X I/O-to-memory transfer

Bits 11-15 DMA I/O read/write command recovery time

This field defines the recovery cycles between two read/write commands.

Bits 16-20 DMA I/O read/write command wait state

This field defines the I/O read/write command wait state.

Bit 21 Set DACK0 low active

0 Set DMA acknowledge signal DACK0 to high active

1 Set DMA acknowledge signal DACK0 to low active

Bit 22 Set DACK1 low active

0 Set DMA acknowledge signal DACK1 to high active

1 Set DMA acknowledge signal DACK1 to low active

Bit 23 Set CS0 low active

0 Set I/O device chip select CS0 to high active

1 Set I/O device chip select CS0 to low active

Bit 24 Set CS1 low active

0 Set I/O device chip select CS1 to high active

1 Set I/O device chip select CS1 to low active

Bit 25 DACK1 active

1 Indicate DMA channel 1 acknowledge DACK1 is active

Bit 26 Channel 0 time out (read only)

1 Indicate channel 0 IORDY signal timeout

Bit 27 Channel 1 time out (read only)

1 Indicate channel 1 IORDY signal timeout

Bits 28-31 Reserved

DMA I/O Device 0 Base Address Register (DBA0)

Port address: 0xf0000220 **Access type**: read/write **Default**: 0xfffff000

0	1	2	3	4	5	6	7
I/O Device 0 Base Address [0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
I/O Device 0 Base Address [8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
I/O Device 0 Base Address [16:19]				Reserved			
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved							

Bits 0-19 DMA I/O device 0 base address

Define DMA device 0 I/O base address. The base address should not conflict to internal mega cell base, and the bits 0-3 should be always set to 0.

Bits 20-31 Reserved

DMA I/O Device 1 Base Address Register (DBA1)

Port address: 0xf0000224 **Access type**: read/write **Default**: 0xfffff000

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
I/O Device 1 Base Address [0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
I/O Device 1 Base Address [8:15]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
I/O Device 1 Base Address [16:19]				Reserved			
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Reserved							

Bits 0-19 DMA I/O device 1 base address

Define DMA device 1 I/O base address. The base address should not conflict to internal mega cell base, and the bits 0-3 should be always set to 0.

Bits 20-31 Reserved

Length Counter 0 Register (LCAR0)

Port address: 0xf0000228 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							LENC0
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
LENC1-8							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
LENC9-16							

Bits 0-14 Reserved

Bits 15-31 Length counter 0

Length counter indicates the remainder to be transfer. DMA transferred number = Length Register - LENC. TC is asserted by Length counter reaching 0. Reading Length Counter may not get the valid value if the channel is active, for the length may be in transition.

Length Counter 0 Register (LCAR0)

Port address: 0xf000022c **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							LENC0
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
LENC1-8							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
LENC9-16							

Bits 0-14 Reserved

Bits 15-31 Length counter 1

Length counter indicates the remainder to be transfer. DMA transferred number = Length Register - LENC. TC is asserted by Length counter reaching 0. Reading Length Counter may not get the valid value if the channel is active, for the length may be in transition.

6.4 PCI BRIDGE INTERFACE REGISTERS

There are four 32-bit registers included in the PCI Bridge Interface controller. The IO address map is allocated from 0xf0000250 to 0xf000025c.

Port Address	Symbol	Access	Description
BA + 0x250	REG0	R/W	Master 0 latency register
BA + 0x254	REG1	R/W	Master 1 latency register
BA + 0x258	REG2	R/W	Master 2 latency register
BA + 0x25c	REG3	R/W	Master 3 latency register

Master 0 Latency Register (REG0)

Port address: 0xf0000250 **Access type**: read/write **Default**: 0x000003ff

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
PERRen	SERRen	Reserved			REQ0_reg[0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
REQ0_reg[3:10]							

Bits 0-15 Reserved

Bit 16 Parity error enable

0 Disable

1 Enable

Bit 17 System error enable

0 Disable

1 Enable

Bits 18-20 Reserved

Bits 21-31 Number of PCICLK count for master latency adjustment

Latency Time = REQ0_reg[0:11] / PCICLK

Master 1 Latency Register (REG1)

Port address: 0xf0000254 **Access type**: read/write **Default**: 0x000003ff

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved			CPURST	FIX	REQ1_reg[0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
REQ1_reg[3:10]							

Bits 0-18	Reserved
Bit 19	CPU reset signal
0	Disable
1	Generate CPU reset signal
Bit 20	Request priority select
0	Rotate priority
1	Fix priority
Bits 21-31	Number of PCICLK count for master latency adjustment Latency Time = REQ1_reg[0:11] / PCICLK

Master 1 Latency Register (REG2)

Port address: 0xf0000258 **Access type**: read/write **Default**: 0x000003ff

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved	PCICLK_sel[0:3]				REQ2_reg[0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
REQ2_reg[3:10]							

Bits 0-16 Reserved

Bits 17-20 PCICLK output delay referenced to EXTCLK

PCICLK_sel[0:3]	Output Delay Step
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	10
1010	12
1011	14
1100	16
1101	18
1110	20
1111	25

Bits 21-31 Number of PCICLK count for master latency adjustment

Latency Time = REQ2_reg[0:11] / PCICLK

Master 1 Latency Register (REG3)

Port address: 0xf000025c **Access type**: read/write **Default**: 0x000003ff

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved					REQ3_reg[0:2]		
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
REQ3_reg[3:10]							

Bits 0-20	Reserved
Bits 21-31	Number of PCICLK count for master latency adjustment Latency Time = REQ3_reg[0:11] / PCICLK

6.5 WIO BUS CONTROLLER

The W90221X provides an ISA-like bus, called the WIO bus, for those low speed devices such as ROM, Flash, or other 8/16-bit I/O devices. The bus shares the 32-bit address/data bus, the 4-bit command/byte enable bus, and the INTD# signal with PCI bus. The WIO bus supports only 8-bit memory (like ROM/Flash) and 8- or 16-bit I/O devices attached on it. Also, the WIO bus provides up to 16 Mbytes (consisting of twenty-four address lines) space for memory and 64K (consisting of sixteen address lines) for I/O devices. Two base registers, containing 8-bit base address for memory and 16-bit for I/O, specify the starting address of the WIO memory space and I/O space. The I/O address space 0xf0000360 ~ 0xf0000367 is allocated for three 16-bit configuration registers of the WIO bus controller.

Port Address	Symbol	Access	Description
BA + 0x360	CFG	R/W	Configuration register
BA + 0x362	WIOBASE	R/W	WIO I/O space base register
BA + 0x364	WMBASE	R/W	WIO ROM space base register
BA + 0x366	-	R/W	Reserved

Configuration Register (CFG)

Port address: 0xf0000360 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
IOen	IOext[0:2]			ROMWext[0:2]			ROMRext[0]
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
ROMRext[1:2]		CMDset[0:1]		CMDhold[0:1]		CMDrec[0:1]	

Bit 0

WIO I/O space enable (I/O devices connected on this WIO bus will only be enabled by turning on this bit)

0	I/O space disable
1	I/O space enable

Bits 1-3

I/O read/write command wait state

These three bits define the wait states of I/O read or write commands appeared on the WIO bus. The I/O read or write commands will be active for "CFG[1:3] + 2" PCICLK cycles.

Bits 4-6

Memory (Flash) write command wait state

These three bits define the wait states of memory write commands appeared on the WIO bus. The memory write commands will be active for "CFG[4:6] + 2" PCICLK cycles.

Bits 7-9

Memory (ROM/Flash) read command wait state

These three bits define the wait states of memory read commands appeared on the WIO bus. The memory read commands will be active for "CFG[7:9] + 2" PCICLK cycles.

Bits 10-11

Command setup time

These two bits define the address-to-command setup time of all command cycles as well as the data-to-command setup time of all write cycles. The setup time is "CFG[10:11] + 2" PCICLK cycles immediately preceding the falling edge of RD_/WR_ signals.

Bit 12-13

Command hold time

These two bits define the address/data-to-command hold time of write cycles. The hold time is "CFG[12:13] + 2" PCICLK cycles immediately following the rising edge of WR_ signal. As for read cycles, the minimum data hold time requirement is zero.

Bits 14-15

Consecutive ROM read command recovery time

For Flash/ROM read cycles, the WIO supports only 32-bit access. The WIO controller will automatically convert the memory word access into four consecutive byte accesses, and the latency between consecutive RD_ cycles will be "CFG[14:15] + 2" PCICLK cycles.

WIO I/O Base Register (WIOBASE)

Port address: 0xf0000362 **Access type**: read/write **Default**: 0x0

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
WIOBASE[0:7]							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
WIOBASE[8:15]							

Bits 0-15

WIO I/O space base address

This register defines the starting address of WIO I/O space on a 64K boundary. As the high halfword (16 bits) address lines of PCI accesses match the WIOBASE[0:15], and if CFG[0] is set high, WIO controller responds DEVSEL_ and TRDY_ to PCI bridge and issues an I/O access cycle to WIO I/O devices.

WIO Memory Base Register (WMBASE)

Port address: 0xf0000364 **Access type**: read/write **Default**: 0x1ef

0	1	2	3	4	5	6	7
Reserved							RAWS
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
WMBASE[0:7]							

Bit 7

Always ROM cycle

Upon setting this bit to logic one, all following PCI accesses will be redirected to WIO memory (ROM/Flash) accesses. This bit is set after each cold start so that the chip's initialization (ROM access) will be redirected to WIO bus where the code ROM attached. Turn off this bit once the other PCI or WIO devices needed to be enabled.

Bits 8-15

WIO memory space base address

This register defines the starting address of WIO memory space on a 16M boundary. As the high byte address lines of PCI accesses match the WMBASE[0:7], WIO controller responds DEVSEL_ and TRDY_ to PCI bridge and issues a memory access cycle to WIO memory devices.

6.6 Parallel Port Interface Registers

There are eleven registers included in the Parallel Port Interface (PPI) controller. The I/O address map is allocated from 0xf0000370 to 0xf000037f.

Port Address	Symbol	Access	Description
BA + 0x378	DLR	R/W	Data Line Register
BA + 0x379	DSR	R	Device Status Register
BA + 0x37a	DCR	R/W	Device Control Register
BA + 0x37b	FSR	R	FIFO Status Register
BA + 0x37c	FCR	R/W	FIFO Control Register
BA + 0x37d	IER	R/W	Interrupt Enable Register
BA + 0x37e	IIR	R	Interrupt Identification Register
BA + 0x37f	DR	R	Data Register
BA + 0x370	DFIFO	R/W	Data FIFO
BA + 0x374	CMD	R/W	Command Register
BA + 0x375	TOR	R/W	Time Out Register
BA + 0x376 ~ BA + 0x377	-	-	Reserve for PPI future extension

Data Line Register (DLR)

Port address: 0xf0000378 **Access type**: read **Default**: 0x---

0	1	2	3	4	5	6	7
8-bit Data Line Status							

Bits 0-7 Data line status

This is the standard parallel port data register. Writing to this register will drive data to the parallel port data lines. Reading from this register will return the value on these data lines.

Device Status Register (DSR)

Port address: 0xf0000379 **Access type**: read **Default**: 0x---

0	1	2	3	4	5	6	7
BUSY#	NACK	PE	SEL	NFAULT	EMPTY	FULL	CMDTRUE

Bit 0	Inverted parallel port interface signal “Busy”
Bit 1	Parallel port interface signal “nAck”
Bit 2	Parallel port interface signal “Perror”
Bit 3	Parallel port interface signal “Select”
Bit 4	Parallel port interface signal “nFault”
Bit 5	Echo device data FIFO empty status
0	Device data FIFO is not empty
1	Device data FIFO is empty
Bit 6	Echo device data FIFO full status
0	Device data FIFO is not full
1	Device data FIFO is full
Bit 7	Command pended
0	Command register (CMD) contains no command code
1	A command code is in CMD which has not been transferred yet.

This read-only register reflects the inputs on the Parallel Port Interface and some of device data FIFO and Command Register status.

Device Control Register (DCR)

Port address: 0xf000037a **Access type**: read **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved		DOE	NACK_IEN	NSELIN#	NINIT	NAUFD#	NSTB#

Bits 0-1 Reserved

Bit 2 Data bus output enable

0 Data bus is driven by PPI for forward transferring

1 Data bus is driven by peripheral device for reverse transferring

Bit 3 nAck interrupt enable

When this bit is set, a low-to-high transition on nAck line will generate an interrupt request to CPU core.

Bit 3 Parallel port interface signal “Select”

Bit 4 Inverted parallel port interface signal “nSelectIn”

Bit 5 Parallel port interface signal “nInit”

Bit 6 Inverted parallel port interface signal “nAutoFd”

Bit 7 Inverted parallel port interface signal “nStrobe”

This register directly controls several output signals as well as enabling some functions. The power-on default "0x0" sets {nSelectIn, nInit, nAutoFd, nStrobe} to {high, low, high, high} states, and enables 8-bit data bus in output mode which are suit for "standard mode" transferring.

FIFO Status Register (FSR)

Port address: 0xf000037b **Access type**: read **Default**: 0x---

0	1	2	3	4	5	6	7
FIFO valid bytes					DA	SA	OV

Bits 0-4 Valid bytes in device data FIFO (DFIFO)

During forward transferring, these bits indicate how many bytes in 16-byte register DFIFO are still not transferred yet. While during reverse transferring, these bits indicate the number of data bytes that are received from the parallel port interface and not read by CPU core.

Bit 5 DFIFO data available

0 DFIFO contains data bytes less than one “Pword”

1 DFIFO contains at least one valid “Pword”

Bit 6 DFIFO space available

0 DFIFO contains empty locations less than one “Pword”

1 DFIFO contains at least one empty “Pword” location

Bit 3 DFIFO over/under run

0 DFIFO is not yet over/under run

1 DFIFO is already over/under run (Once this bit is set, it will remain set until DFIFO or the PPI is reset)

FIFO Control Register (FCR)

Port address: 0xf000037c **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
DMAen	FRST	DRST	PWord	MOD		RDTH	

Bit 0

DMA mode enable

A low-to-high transition on this bit will make PPI issue a DREQ to DMA controller. Upon receiving the corresponding DACK, PPI deasserts the DREQ. This bit will be cleared by DMA terminal-count (TC) asserting or by a CPU write cycle with data-in[0] = 0.

Bit 1

Reset DFIFO

Writing a logical one to this bit will assert "DFIFO Reset" for one EXTCLK cycle. This bit will return to deasserted state automatically after "DFIFO Reset" is issued.

Bit 2

Reset device

Writing a logical one to this bit will assert "Device Reset" for one EXTCLK cycle. This bit will return to deasserted state automatically after "Device Reset" is issued.

Bit 3

PWord (PWord defines the basic unit of DFIFO access during CPU cycle)

0	PWord is 8-bit (one byte)
1	PWord is 32-bit (four bytes)

Bits 4-5

Device mode select

IER[1] and FCR[4:5] are used to choose device operation mode

{IER[1], FCR[4:5]}	Device Operation Mode
1X0	Test Mode
1X1	Peripheral Emulation Mode
000	Standard Mode
001	PS2 Mode
010	Fast Standard Mode
011	ECP Mode

Bits 6-7

DFIFO read threshold

These two bits define the threshold level for triggering data-available interrupt (Irpt_RDA) of DFIFO during reverse transferring.

FCR[6:7]	Read Threshold Level	
	PWord = 1 byte	PWord = 4 bytes
00	16 bytes	16 bytes
01	12 bytes	12 bytes
10	8 bytes	8 bytes
11	1 byte	4 bytes

Interrupt Enable Register (IER)

Port address: 0xf000037d **Access type**: read/write **Default**: 0x0

0	1	2	3	4	5	6	7
Reserved	PEMU	Tout_Ien	TC_Ien	Temp_Ien	Rda_Ien	nFault_Ien	LOOP

Bit 0 Reserved

Bit 1 Peripheral emulation mode enable

0 Device is not operating in "peripheral emulation mode" or "test mode"

1 Set device to "peripheral emulation mode" or "test mode"

(This bit along with FCR[4:5] are used to choose device operation mode.)

Bit 2 Time-out interrupt (Irpt_TOUT) enable

0 Mask Irpt_TOUT

1 Enable Irpt_TOUT

Bit 3 DMA terminal-count interrupt (Irpt_TC) enable

0 Mask Irpt_TC

1 Enable Irpt_TC

Bit 4 DFIFO empty interrupt (Irpt_TEMP) enable

0 Mask Irpt_TEMP

1 Enable Irpt_TEMP

Bit 5 DFIFO read threshold interrupt (Irpt_RDA) enable

0 Mask Irpt_RDA

1 Enable Irpt_RDA

Bit 6 nFault Interrupt (Irpt_nFault) enable

0 Mask Irpt_nFault

1 Enable Irpt_nFault

Bit 7 Loop back enable

0 Loop-back disable

1 Loop-back enable

(During loop-back mode, {nStrobe, nAutoFd, nInit, nSelectIn} will be fed to {nAck, Busy, PError, nFault} internally. This mode is used only for test issue.)

Interrupt Identification Register (IIR)

Port address: 0xf000037e **Access type**: read **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved		Irpt_TOUT	Irpt_TC	Irpt_TEMP	Irpt_RDA	Irpt_nFault	Irpt_nAck

Bits 0-1 Reserved

Bit 2 Time-out interrupt flag

0 Reset device, CPU reads Time-Out Register (TOR), or DFIFO being accessed either by CPU or parallel port interface transferring

1 If IER[2] is set, and "Time out" is occurred during parallel port transferring.

Bit 3 DMA terminal count interrupt flag

0 TC is deasserted by DMA controller.

1 If IER[3] is set, and TC is asserted by DMA controller once DMA transfer is done.

Bit 4 DFIFO empty interrupt flag

0 CPU writes new data into DFIFO.

1 If IER[4] is set, and DFIFO is empty during "forward transferring".

Bit 5 DFIFO read threshold interrupt flag

0 CPU reads DFIFO such that data bytes in DFIFO are below the threshold level.

1 If IER[5] is set, and data bytes received by DFIFO are exceeded the threshold level (defined in FCR[4:5]) during "reverse transferring".

Bit 6 nFault interrupt flag

0 CPU reads device status register (DSR).

1 If IER[6] is set, and a high-to-low transition is on "nFault" pin.

Bit 7 nAck interrupt flag

0 CPU reads device status register (DSR).

1 If DCR[3] is set, and a low-to-high transition is on "nAck" pin.

Data Register (DR)

Port address: 0xf000037f **Access type**: read **Default**: 0x---

0	1	2	3	4	5	6	7
8-bit Data of Latched Lines Status							

Bits 0-7 Latched line status

The status of data lines of PPI will be latched into this register if a high-to-low transition is happened on "nAck" pin. This register is added to support "Peripheral Emulation Mode" operation.

Device Data FIFO (DFIFO)

Port address: 0xf0000374 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
DFIFO MSB Byte							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
DFIFO LSB Byte							

Bits 0-31 Device data FIFO

The device build-in a 16-byte data FIFO to accelerate the transfer rate when using "Fast Standard Mode" or "ECP mode". The DFIFO may be 1-byte or 4-byte accessed by CPU using "PWord" basis.

Command Register (CMD)

Port address: 0xf0000374 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Pending command code							

Bits 0-7 Pending command code

Whenever a command code is written by CPU, a "command transfer" will be induced immediately during "ECP forward transferring". If CMD contains a command code not been transferred yet, a "command pended" status (CMDtrue) is echoed in DSR[7]. "Reset device" or "CPU read CMD" or the pended command code is finished transferring, the CMDtrue will also be cleared.

Time-out Register (TOR)

Port address: 0xf0000375 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
TOUTen	TOUTcmp[0:6]						

Bit 0	Time-out counter enable
0	Disable time-out counter
1	Enable time-out counter

Bits 0-7 Time-out counter comparison value

If "TOUTen" is set, the "TOUTcnt[0:6] will be reset first and then start counting whenever a new PPI transfer cycle is initiated. On detecting TOUTcnt[0:6] is equal to TOUTcmp[1:7], a "Time Out" flag will be set which in turn trigger a interrupt request (Irpt_TOUT) if IER[2] (ToutIen) is also set at that time. The tick of Time-Out Counter is about 9.175ms (OSC/(2**21) where OSC = 14.318MHz). The maximum duration that Time-Out Counter can cover is about 1.17 sec (2**7 ticks).

6.7 COM PORT INTERFACE REGISTERS

W90221X Provides 2 COM ports to interface external RS232 devices. COM0 allocates 0xf00003f8
~ oxf00003ff as its IO-space, while COM1 allocates 0xf00002f8 ~ 0xf00002ff as its IO-space.

Table 6.7-1 : COM0 Register Map (IO base (BA) : 0xf0000000)

Port Addr.	Symbol	Acces s	Description
BA + 0x3f8, DLAB = 0	RBR[0:7]	R	Receiver Buffer Register
BA + 0x3f8, DLAB = 0	THR[0:7]	W	Transmitter Holding Register
BA + 0x3f9, DLAB = 1	IER[3:7]	R/W	Interrupt Enable Register
BA + 0x3f8, DLAB = 1	DLL[0:7]	R/W	Divisor Latch Register (LS)
BA + 0x3f9, DLAB = 1	DLM[0:7]	R/W	Divisor Latch Register (MS)
BA + 0x3fa	IIR[0:7]	R	Interrupt Identification Register
BA + 0x3fa	FCR[0:7]	W	FIFO Control Register
BA+ 0x3fb	LCR[0:7]	R/W	Line Control Register
BA + 0x3fc	MCR[0:7]	R/W	Modem Control Register
BA + 0x3fd	LSR[0:7]	R	Line Status Register
BA + 0x3fe	MSR[0:7]	R	MODEM Status Register
BA + 0x3ff	TOR[0:7]	R/W	Time Out Register

Table 6.7-2 : COM1 Register Map (IO base (BA) : 0xf0000000)

Port Addr.	Symbol	Acces s	Description
BA + 0x2f8, DLAB = 0	RBR[0:7]	R	Receiver Buffer Register
BA + 0x2f8, DLAB = 0	THR[0:7]	W	Transmitter Holding Register
BA + 0x2f9,	IER[3:7]	R/W	Interrupt Enable Register

DLAB = 1			
BA + 0x2f8, DLAB = 1	DLL[0:7]	R/W	Divisor Latch Register (LS)
BA + 0x2f9, DLAB = 1	DLM[0:7]	R/W	Divisor Latch Register (MS)
BA + 0x2fa	IIR[0:7]	R	Interrupt Identification Register
BA + 0x2fa	FCR[0:7]	W	FIFO Control Register
BA+ 0x2fb	LCR[0:7]	R/W	Line Control Register
BA + 0x2fc	MCR[0:7]	R/W	Modem Control Register
BA + 0x2fd	LSR[0:7]	R	Line Status Register
BA + 0x2fe	MSR[0:7]	R	MODEM Status Register
BA + 0x2ff	TOR[0:7]	R/W	Time Out Register

Receiver Buffer Register (RBR)

Port address : 0xf00003f8, DLAB=0 (COM0)   Read only   Power-on Default : --
0xf00002f8, DLAB=0 (COM1)

0	1	2	3	4	5	6	7
8-bit Receiver Data							

Bits 0-7   Receiver Data

Reading this register, COM port returns 8-bit data receiving from SDI pin.

Transmitter Holding Register (THR)

Port address : 0xf00002f8, DLAB=0 (COM0)   Write only   Power-on Default : --
0xf00002f8, DLAB=0 (COM1)

0	1	2	3	4	5	6	7
8-bit Transmit Data							

Bits 0-7 Transmit Data

Writing to this register, COM port will sent out the data through SDO pin (THR[7] first).

Interrupt Enable Register (IER)

Port address : 0xf00003f9, DLAB=0 (COM0) Read/Write Power-on Default : 0x0

0xf00002f9, DLAB=0 (COM1)

0	1	2	3	4	5	6	7
				MOS_Ien	RLS_Ien	THRE_Ien	RDA_Ien

Bit 4 MODEM Status Interrupt (Irpt_MOS) Enable

0 = Mask Irpt_MOS

1 = Enable Irpt_MOS

Bit 5 Receiver Line Status Interrupt (Irpt_RLS) Enable

0 = Mask Irpt_RLS

1 = Enable Irpt_RLS

Bits 6 Transmitter Holding Register Empty Interrupt (Irpt_THRE) Enable

0 = Mask Irpt_THRE

1 = Enable Irpt_THRE

Bits 7 Receiver Data Available Interrupt (Irpt_RDA) and Time-Out Interrupt (Irpt_TOUT) Enable

0 = Mask Irpt_RDA and Irpt_TOUT

1 = Enable Irpt_RDA and Irpt_TOUT

Divisor Latch (low byte) Register (DLL)

Port address : 0xf00003f8, DLAB=1 (COM0) Read/write Power-on Default : 0x0

0xf00002f8, DLAB=1 (COM1)

0	1	2	3	4	5	6	7

Baud Rate Divisor (Low Byte)

Bit 0-7 Low byte of baud rate dvisor

Divisor Latch (high byte) Register (DLM)

Port address : 0xf00003f9, DLAB=1 (COM0)   Read/write   Power-on Default : 0x0
0xf00002f9, DLAB=1 (COM1)

0	1	2	3	4	5	6	7
Baud Rate Divisor (High Byte)							

Bit 0-7 High byte of baud rate dvisor

The 16-bit Divisor ({DLM, DLL}) is used to determine the COM port's baud rate. The equation is

Baud Rate = Frequency input / {16 * [Divisor +2]}

Interrupt Identification Register (IIR)

Port address : 0xf00003fa (COM0)   Read only   Power-on Default : ---
0xf00002fa (COM1)

0	1	2	3	4	5	6	7
FMENo	RTHo[0:1]		DMOD	IID[0:2]			NOI

Bits 0 Status of "FIFO Mode Enable"
This bit echos if "FIFO mode" is enable or not. Since "FIFO mode" is always enable,
this bit always shows logical 1 when CPU reading this register.

Bit 1-2 Status of RX FIFO threshold level
These bits show current setting of receiver FIFO threshold level (RTH). The meaning of
RTH is defined in the following FCR description.

Bit 3 DMA mode select

The DMA function is **not implemented** in this version. Reading IIR, the bit-3 is always 0.

Bit 4-6 Interrupt Identification bits

The IID[0:2] along with NOI indicate current interrupt request from COM port

Bit 7 No Interrupt (NOI) pended

Table 6.7-3 : Interrupt Control Functions

IIR[4:7]	Priority	Interrupt Type	Interrupt Source	Interrupt Reset control
- - - 1	--	None	None	--
0 1 1 0	Highest	Receiver Line Status (Irpt_RLS)	Overrun Error or Parity Error or Framing Error or Break Interrupt	Reading the LSR
0 1 0 0	Second	Received Data Available (Irpt_RDA)	Receiver FIFO thres-hold level is reached	Receiver FIFO drops below the threshold level
1 1 0 0	Second	Receiver FIFO Time-out (Irpt_TOUT)	Receiver FIFO is non-empty and no activities are occured in receiver FIFO during the TOR defined time duration	Reading the RBR
0 0 1 0	Third	Transmitter Hoding Register Empty (Irpt_THRE)	Transmitter Holding Register Empty	Reading the IIR (if source of interrupt is Irpt_THRE) or writing into the THR
0 0 0 0	Fourth	MODEM Status (Irpt_MOS)	CTS, DSR, DCD bits chang state or RI bit changes from high to low	Reading the MSR

FIFO Control Register (FCR)

Port address : 0xf00003fa (COM0)   Write only   Power-on
Default : 0x1
0xf00002fa (COM1)

0	1	2	3	4	5	6	7
RTH[0:1]		Reserved		DMOD	TXRST	RXRST	FMEN

Bits 0-1   RX FIFO interrupt (Irpt_RDA) trigger level

FCR[0:1]	Irpt_RDA trigger level (bytes)
0 0	01
0 1	04
1 0	08
1 1	14

Bit 4   DMA mode select

The DMA function is **not implemented** in this version.

Bit 5   Reset TX FIFO

Seting this bit will generate 1 OSC cycle reset pulse to reset TX FIFO. The TX FIFO becomes empty (TX-pointer is cleared to 0) after such reset. This bit is returned to 0 automatically after the reset pulse is generated.

Bit 6   Reset RX FIFO

Seting this bit will generate 1 OSC cycle reset pulse to reset RX FIFO. The RX FIFO becomes empty (RX-pointer is cleared to 0) after such reset. This bit is returned to 0 automatically after the reset pulse is generated.

Bit 7   FIFO mode enable

The UART0 and UART1 are always operated on FIFO mode. Writing this bit has no effect while reading this bit always get logical one.

Line Control Register (LCR)

Port address : 0xf00003fb (COM0) Read/Write Power-on Default : 0x0
0xf00002fb (COM1)

0	1	2	3	4	5	6	7
DLAB	BREAK	SPAR	EPAR	PAR	STOP	WLEN	

Bits 0 Divisor Latch Access Bit

0 = "2F8/3F8" and "2F9/3F9" are used to access RBR, THR or IER.

1 = "2F8/3F8" and "2F9/3F9" are used to access Divisor Latch Registers (DLL, DLM).

Bit 1 Break Control Bit

When this bit is set to a logic 1, the serial data output (SOUT) is forced to the **Spacing State**

(logic 0). This bit affects SOUT only and has no effect on the transmitter logic.

Bit 2 Stick Parity Enable

0 = Disable Stick Parity

1 = The parity bit is transmitted and checked as a logic 1 if bit-3=0 (odd parity), or

as a logic 0 if bit-3=1 (even parity).

This bit has effects only when bit-4 (Parity Bit Enable) is set.

Bit 3 Even Parity Enable

0 = Odd number of logic 1s is transmitted or checked in the data word bits and parity bit.

1 = Even number of logic 1s is transmitted or checked in the data word bits and parity bit.

This bit has effects only when bit-4 (Parity Bit Enable) is set.

Bit 4 Parity Bit Enable

0 = Praity bit is not generated (transmit data) or checked (receive data) during transfer.

1 = Parity bit is generated of checked between the "last data word bit" and "stop bit" of

the serial data.

Bit 5 Number of "Stop bit"

0 = One "stop bit" is generated in the transmitted data.

1 = **One and a half** "stop bit" is generated in the transmitted data when **5-bit** word length
is selected.
Two "stop bit" is generated when **6-, 7-** and **8-bit** word length is selected.

Bits 6-7 Word Length Select

LCR[6:7]	Character length
0 0	5 bits
0 1	6 bits
1 0	7 bits
1 1	8 bits

Modem Control Register (MCR)

Port address : 0xf00003fc (COM0) Read/Write Power-on Default : 0x0
0xf00002fc (COM1)

0	1	2	3	4	5	6	7
Reserved			LOOP	OUT2#	OUT1#	RTS#	DTR#

Bits 3 Enable Loop-Back mode

0 = Disable

1 = When loop-back is enable, the following signals is connected internally.

SOUT connects to **SIN** internally and SOUT pin is fixed logic 1.

DTR# connects to **DSR#** internally and DTR# pin is fixed logic 1.

RTS# connects to **CTS#** internally and RTS# pin is fixed logic 1.

OUT1# connects to **RI#** internally and OUT1# pin is fixed logic 1.

OUT2# connects to **DCD#** internally and OUT2# pin is fixed logic 1.

Bit 4 Complement version of OUT2# (user-designated output) signal

Bit 5 Complement version of OUT1# (user-designated output) signal

Bit 6 Complement version of RTS# (Request-To-Send) signal

Bit 7 Complement version of DTR# (Data-Terminal-Ready) signal

Writing 0x00 to MCR **set** DTR#, RTS#, OUT1# and OUT2# to logic 1s, while writing 0x0f to MCR **reset** DTR#, RTS#, OUT1# and OUT2# to logic 0s.

Line Status Register (LSR)

Port address : 0xf00003fd (COM0) Read only Power-on Default : ---

0xf00002fd (COM1)

0	1	2	3	4	5	6	7
Err_RCVR	TEMT	THRE	BI	FE	PE	OE	DR

Bits 0 RX FIFO Error

0 = RX FIFO works normally

1 = There is at least one parity error (PE), framing error (FE) or break indication (BI)

in the FIFO. LSR[0] is cleared when CPU reads the LSR and if there are no sub-

sequent errors in the RX FIFO.

Bit 1 Transmitter Empty

0 = Either Transmitter Holding Register (**THR** - TX FIFO) or Transmitter Shift Register

(**TSR**) are not empty.

1 = Both THR and TSR are empty.

Bit 2 Transmitter Holding Register Empty

0 = THR is not empty.

1 = THR is empty.

The THRE bit is set when the last data word of TX FIFO is transferred to TSR. This bit is reset

concurrently with the loading of the THR (or TX FIFO) by the CPU. This bit also causes the

UART to issue an interrupt (Irpt_THRE) to the CPU when IER[6]=1.

Bit 3 Break Interrupt indicator

This bit is set to a logic 1 whenever the received data input is held in the "spacing state"

(logic 0) for longer than a full word transmission time (that is, the total time of "start bit"

+ data bits + parity + stop bits).

Bit 4 Framing Error indicator

This bit is set to a logic 1 whenever the received character did not have a valid "stop bit"

(that is, the stop bit following the last data bit or parity bit is detected as a logic 0).

Bit 5 Parity Error indicator

This bit is set to a logic 1 whenever the received character did not have a valid "parity bit".

Bit 6 Overrun Error indicator

An overrun error will occur only after the RX FIFO is full and the next character has been

completely received in the shift register. The ccharacter in the shift register is overwritten,

but it is not transferred to the RX FIFO. OE is indicated to the CPU as soon as it happens

and is reset whenever the CPU reads the contents of the LSR.

Bit 7 RX FIFO Data Ready

0 = RX FIFO is empty

1 = RX FIFO contains at least 1 received data word.

LSR[3:5] (BI, FE, PE) is revealed to the CPU when its associated character is at the top of the RX

FIFO. These three error indicators are reset whenever the CPU reads the contents of the LSR.

LSR[3:6] (BI, FE, PE, OE) are the error conditions that produce a "receiver line status interrupt"

(Irpt_RLS) when IER[5]=1. Read LSR clear Irpt_RLS.

Writing LSR is a null operation (not suggested).

Modem Status Register (MSR)

Port address : 0xf00003fe (COM0)   Read only   Power-on
Default : ---
0xf00002fe (COM1)

0	1	2	3	4	5	6	7
DCD#	RI#	DSR#	CTS#	DDCD	TERI	DDSR	DCTS

Bits 0 Complement version of **Data Carrier Detect** (DCD#) input

Bits 1 Complement version of **Ring Indicator** (RI#) input

Bits 2 Complement version of **Data Set Ready** (DSR#) input

Bits 3 Complement version of **Clear to Send** (CTS#) input

Bits 4 DCD# state change

This bit is set whenever DCD# input has changed state, and it will be reset if the CPU reads
the MSR.

Bits 5 Tailing edge of RI

This bit is set whenever TI# input has changed from high to low, and it will be reset if the
CPU reads the MSR.

Bits 6 DSR# state change

This bit is set whenever DSR# input has changed state, and it will be reset if the CPU reads
the MSR.

Bits 7 CTS# state change

This bit is set whenever CTS# input has changed state, and it will be reset if the CPU reads
the MSR.

Whenever either of MSR[4:7] is set to logic 1, a Modem Status Interrupt is generated if IER[4]=1.
Writing LSR is a null operation (not suggested).

Time Out Register (TOR)

Port address : 0xf00003ff (COM0)   Read/Write Power-on Default : 0x0
0xf00002ff (COM1)

0	1	2	3	4	5	6	7
TOUT_ en	TOUT_cmp						

Bits 0 Time-Out (interrupt) enable

The feature of Receiver Time-Out (interrupt) is enable only when TOR[0] = IER[7] = 1.

Bits 1-7   Time-Out (interrupt) comparator

The Time-Out counter is reset and start counting (the counting clock = baud rate) whenever

the RX FIFO receives a new data word. Once the content of Time-Out counter (TOUT_cnt)

is equal to that of Time-Out comparator (TOUT_cmp), a Receiver Time-Out interrupt Irpt_

TOUT) is generated if TOR[0] = IER[7] = 1.

A new incoming data word or RX FIFO empty clears Irpt_TOUT.

6.8 Synchronous Serial Interface Registers

There are five registers included in the Synchronous Serial Interface (SSI) controller. The I/O address map is allocated from 0xf0000380 to 0xf000038a.

Port Address	Symbol	Access	Description
BA + 0x380	DFIFO	R/W	Data FIFO
BA + 0x384	CFGH	R/W	High Configuration Register
BA + 0x386	CFGL	R/W	Low Configuration Register
BA + 0x388	CTRL	R/W	Control Register

Data FIFO Register (DFIFO)

Port address: 0xf0000380 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							RAWS
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
WMBASE[0:7]							

Bit 7

Always ROM cycle

Upon setting this bit to logic one, all following PCI accesses will be redirected to WIO memory (ROM/Flash) accesses. This bit is set after each cold start so that the chip's initialization (ROM access) will be redirected to WIO bus where the code ROM attached. Turn off this bit once the other PCI or WIO devices needed to be enabled.

Bits 8-15

WIO memory space base address

This register defines the starting address of WIO memory space on a 16M boundary. As the high byte address lines of PCI accesses match the WMBASE[0:7], WIO controller responds DEVSEL_ and TRDY_ to PCI bridge and issues a memory access cycle to WIO memory devices.

Data FIFO Register (Dfifo)

Port address : 0xf0000380 Read/Write Power-on Default : --

0	1	2	3	4	5	6	7
Dfifo MSB Byte							

8	9	10	11	12	13	14	15

16	17	18	19	20	21	22	23

24	25	26	27	28	29	30	31
Dfifo LSB byte							

The device build-in a 48x16 or 24x32 data fifo to accelerate the transfer rate.

The Dfifo may be 16-bit or 32-bit accessed by CPU, the type of reading, 16-bit or 32-bit depends on

RX_FIFO type, and the type of writing, 16-bit or 32-bit depends on TX_FIFO type.

Bits 0-31

PCM data in/out, Whether the MSB bits are sign- or zero-extension depends on MEXT (CFGH[6]).

High Configuration Register (CFGH)

Port address : 0xf0000384 Read/Write Power-on Default : 0x0000

0	1	2	3	4	5	6	7
SSIEN	LOOP	MASTER	LFMOD	Reserved	FACT	MEXT	SLEN[0]

8	9	10	11	12	13	14	15
SLEN[1:4]				WPF[0:3]			

Bits 0 SSI Enable
0 = SSI disable
1 = SSI enable

Bit 1 Loop back enable
0 = disable
1 = enable

Bit 2 SYNC master mode enable
0 = SYNC is input.
1 = SYNC is output.

Bit 3 Long Framing Mode
0 = Short framing
The first data will be available on the next SCLK cycle as SYNC is active. In this mode,
SYNC width is 1 SCLK.
1 = Long framing
The first data will be available on the same SCLK cycle as SYNC is active. In this mode,
SYNC width is 1 SLEN.

Bit 4 Reserved

Bit 5 Frame active level
0 = active high
1 = active low

Bit 6 RX-FIFO MSB extension
0 = fill 0 in redundant MSBs of RX-FIFO
1 = non-implement

Bit 7-11 Serial word length
Word length = SLEN[0:4] + 1.
The word length supported by SSI is from 1 to 32 bits.
SLEN[0:4] configure TX/RX also.
if SLEN[0:4] <= 15, FIFO will be configured as 48x16.

if 15 < SLEN[0:4] <= 31, FIFO will be configured as 24x32.

Bit 12-15 Words per Frame
Words per frame = WPF[0:3] + 1. (max. 16 words/frame)

Low Configuration Register (CFGL)

Port address : 0xf0000386 Read/write Power-on Default : 0x0000

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td colspan="8">BPF[0:7]</td></tr>
</table>

<table>
<tr><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
<tr><td colspan="8">SCLKDIV[0:7]</td></tr>
</table>

Bit 0-7 Number of bits per frame
Bits per frame = BPF[0:7] + 1. (max. 256 bits/frame)

Bit 8-15 Serial clock divider
On master mode,the SCLK is an output and its frequency is
SCLK frequency = EXTCLK / (2*(SCLKDIV+1))

Control Register (CTRL)

Port address : 0xf0000388 Read/Write Power-on Default : 0x0000

<table>
<tr><td>0</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td></tr>
<tr><td>DVRST</td><td>TXRST</td><td>RXRST</td><td colspan="2">RXTH[0:1]</td><td colspan="2">TXTH[0:1]</td><td>IntRxen</td></tr>
</table>

<table>
<tr><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td></tr>
<tr><td>IntTXen</td><td>IntERRen</td><td colspan="6">Reserved</td></tr>
</table>

Bits 0 Device reset
This is a self-clear bit, ie. set this bit to 1, it will be clear to 0 automatically after 1 EXTCLK.

When this bit is set, all registers will be set to its default value and the controller will be also

set to its initial states.

Bit 1 Reset TX-FIFO

This is a self-clear bit, ie. set this bit to 1, it will be clear to 0 automatically after 1 EXTCLK.

When this bit is set, The TX-FIFO pointer will be cleared to 0, the TX-FIFO is empty

immediately.

Bit 2 Reset RX-FIFO

This is a self-clear bit, ie. set this bit to 1, it will be clear to 0 automatically after 1 EXTCLK.

When this bit is set, The RX-FIFO pointer will be cleared to 0, the RX-FIFO is empty

immediately.

Bit 3-4 RX-FIFO threshold level

00 = RX-FIFO full

01 = 3/4 RX-FIFO

10 = 1/2 RX-FIFO

11 = RX-FIFO non-empty

Bit 5-6 TX-FIFO threshold level

00 = TX-FIFO empty

01 = 1/4 TX-FIFO

10 = 1/2 TX-FIFO

11 = TX-FIFO non-full

Bit 7 RX-FIFO interrupt enable

0 = disable

1 = enable

Bit 8 TX-FIFO interrupt enable

0 = disable

1 = enable

Bit 9 RX-FIFO overrun interrupt enable

0 = disable

1 = enable

Bit 10-15 Reserved

Status Register (STUS)

Port address : 0xf000038a   Read/Write   Power-on Default : --

0	1	2	3	4	5	6	7
RXDA	TXSA	RXERR	Reserved				IntRX

8	9	10	11	12	13	14	15
IntTX	INTRER R	Reserved					

Bits 0 RX-FIFO data available

0 = There is no valid data word in RX-FIFO.

1 = There is at least one valid data word in RX-FIFO.

Bit 1 TX-FIFO space available

0 = There is no space available in TX-FIFO.

1 = The TX-FIFO can still accept at least one data word.

Bit 2 RX-FIFO overrun

0 = The RX-FIFO works well.

1 = The RX-FIFO is already overrun.

Once the RX-FIFO is overrun, this bit will keep active until RX-FIFO is reset.

Bit 3-6 Reserved

Bit 7 RX-FIFO interrupt request

0 = No RX-FIFO interrupt request

1 = A RX-FIFO interrupt request is pending

Set = Valid data words in RX-FIFO exceeds the threshold level.

Reset = Valid data words in RX-FIFO drops below the threshold level.

Bit 8 TX-FIFO interrupt request

0 = No TX-FIFO interrupt request

1 = A TX-FIFO interrupt request is pending

Set = Valid data words in TX-FIFO drops below the threshold level.

Reset = Valid data words in TX-FIFO exceeds the threshold level

Bit 9 RX-FIFO overrun interrupt request

0 = No RX-FIFO overrun interrupt request

1 = A RX-FIFO overrun interrupt request is pending

Set = When RX-FIFO is overrun.

Reset = Reset RX-FIFO or reset device.

Bit 10-15 Reserved

6.9 Timer Registers

There are four registers included in the timer. The I/O address map is allocated from 0xf0000040 to 0xf0000043.

Port Address	Symbol	Access	Description
BA + 0x40	TCR1	R/W	Timer Control Register 1
BA + 0x41	TICR1	R/W	Timer Initial Control Register 1
BA + 0x42	TCR2	R/W	Timer Control Register 2
BA + 0x43	TICR2	R/W	Timer Initial Control Register 2

Two 24-bit decrementing timers will be implemented, corresponding to the TCR1, TICR1 and TCR2, TICR2 independently. When the timer interrupt enable bit is set to one and the counter decrements to zero, the timer will assert the associated interrupt signal. The interrupt signal will assert one of the thirty-two external interrupts defined by the EI bits in the control register. When a timer reaches zero, the timer hardware reloads the counter with the value from the timer initial count register and continues decrementing.

Timer Control Register 1 (TCR1)

Port address: 0xf0000040 **Access type**: read/write **Default**: 0x---

<table>
<tr><th>0</th><th>1</th><th>2</th><th>3</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>TI</td><td>CE</td><td>IE</td><td colspan="5">Reserved</td></tr>
<tr><td><b>8</b></td><td><b>9</b></td><td><b>10</b></td><td><b>11</b></td><td><b>12</b></td><td><b>13</b></td><td><b>14</b></td><td><b>15</b></td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td><b>16</b></td><td><b>17</b></td><td><b>18</b></td><td><b>19</b></td><td><b>20</b></td><td><b>21</b></td><td><b>22</b></td><td><b>23</b></td></tr>
<tr><td colspan="8">Reserved</td></tr>
<tr><td><b>24</b></td><td><b>25</b></td><td><b>26</b></td><td><b>27</b></td><td><b>28</b></td><td><b>29</b></td><td><b>30</b></td><td><b>31</b></td></tr>
<tr><td colspan="8">Pre-scale Value [0:7]</td></tr>
</table>

Bit 0 Timer interrupt bit

The timer sets this bit to indicate that it has decremented to zero. This bit remains set until the software resets it.

Bit 1 Counter enable bit

Setting the CE bit to one cause the timer to begin decrementing.

Setting the CE bit to zero stops the timer.

Bit 2 Interrupt enable bit

When the IE is set to one and the counter decrements to zero, the timer asserts its interrupt signal to interrupt CPU.

Bits 3-23 Reserved

Bits 24-31 Pre-scale

This pre-scale value is used to divide the input clock.

Timer Initial Control Register 1 (TICR1)

Port address: 0xf0000044 **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Timer Initial Count [0:7]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Timer Initial Count [8:15]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Timer Initial Count [16:27]							

Bits 0-7	Reserved
Bits 8-24	Timer initial count A 24-bit value for timer initial count

Timer Control Register 2 (TCR2)

Port address: 0xf0000048 **Access type**: read/write **Default**: 0x---

0	**1**	**2**	**3**	**4**	**5**	**6**	**7**
TI	CE	IE	Reserved				
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Reserved							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Reserved							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Pre-scale Value [0:7]							

Bit 0	Timer interrupt bit The timer sets this bit to indicate that it has decremented to zero. This bit remains set until the software resets it.
Bit 1	Counter enable bit Setting the CE bit to one cause the timer to begin decrementing. Setting the CE bit to zero stops the timer.
Bit 2	Interrupt enable bit When the IE is set to one and the counter decrements to zero, the timer asserts its interrupt signal to interrupt CPU.
Bits 3-23	Reserved
Bits 24-31	Pre-scale This pre-scale value is used to divide the input clock.

Timer Initial Control Register 2 (TICR2)

Port address: 0xf000004c **Access type**: read/write **Default**: 0x---

0	1	2	3	4	5	6	7
Reserved							
8	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Timer Initial Count [0:7]							
16	**17**	**18**	**19**	**20**	**21**	**22**	**23**
Timer Initial Count [8:15]							
24	**25**	**26**	**27**	**28**	**29**	**30**	**31**
Timer Initial Count [16:27]							

Bits 0-7 Reserved

Bits 8-24 Timer initial count

A 24-bit value for timer initial count

7 ELECTRICAL SPECIFICATIONS

7.1 Absolute Maximum Ratings

Ambient temperature .... 0 °C ~ 70 °C

Storage temperature .... -40 °C ~ 125°C

Voltage on any pin .... Vss-0.5V ~ Vcc+0.5V

Power supply voltage 7V

Injection current (latch-up testing) 100mA

Operating power dissipation 30mA/MHz

7.2 DC Specifications

(Normal test conditions : VDD5V = 5.0V+/- 5%, VDDi/VDDp/VDDl = 3.3V+/-5%, TA = 0 °C ~ 70 °C
unless otherwise specified)

SYMBOL	PARAMETER	CONDITION	MIN	MAX	UNIT
VDD5V	Power Supply		4.75	5.25	V
VDDi/VDDp	Power Supply		3.14	3.46	V
V_{IL}	Input Low Voltage			0.8	V
V_{IH}	Input High Voltage		2.0		V
V_{OL}	Output Low Voltage	I_{OUT} = 2,4,8 mA (*2)		V_{SS}+ 0.4	V
V_{OH}	Output High Voltage	I_{OUT} = -1,2,4 mA (*2)	2.4		V
I_{CC}	Supply Current	F_{cpu} = 100MHz		300	mA
I_{IH}	Input High Current	V_{IN} = 2.4 V (*1)		10	µA
I_{IL}	Input Low Current	V_{IN} = 0.4 V (*1)	-10	10	µA

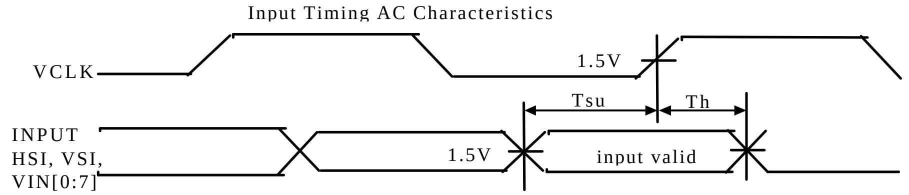
I_{IHP}	Input High Current (pull-up)	V_{IN} = 2.4 V (*3)	-45	-15	µA
I_{ILP}	Input Low Current (pull-up)	V_{IN} = 0.4 V (*3)	-10		µA

Note *1 : Inpt leakage current (I_{IL}, I_{IH}) include those bi-directional pins which are in "input" mode (output disable).

*2 : Pins of **4mA** sink capability include : DTR0n, RTS0n, SOUT0, SOUT0, HSO, VSO, SDO, VD[0:7],
SCLK, SYNC.
Pins of **6mA** sink capability include : PCIRST, PCICLK, GPIO[0:14].
Pins of **8mA** sink capability include : RAS#, CAS#, CKE, CS0H#, CS1H#, CS0L#, CS1L#, WE#,
DQMB[0:3], MA[0:13], GNT0#, GNT1#, INTD#, MD[0:31],
COMBE[0:3], PDA[0:31], X_STOP#, PERR#, FRAME#, IRDY#,
PPAR, TRDY#, DEVSEL#, GPIO[15:16].
Programmable 4/8/16mA sink capability : MCLK.
Current driver of full scale 18.7mA (analog output) : RED, GREEN, BLUE.
External Voltage reference pins : COMP, RSET, VREF.

*3 : Inputs with internal pull-up resistor include : GPIO[17:18], PREQ#[0:1], SERR#, INTA#, INTB#, INTC#,
CTS0n, DSR0n, RI0n, DCD0n, SDI.

7.3 AC Specifications



Symbol	Parameter	Min	Max	Unit
Th	VIN[0:7], HSI, VSI	5		ns
Tsu	VIN[0:7], HSI, VSI	5		ns

Figure 7.3.1 Timing of VMI bus

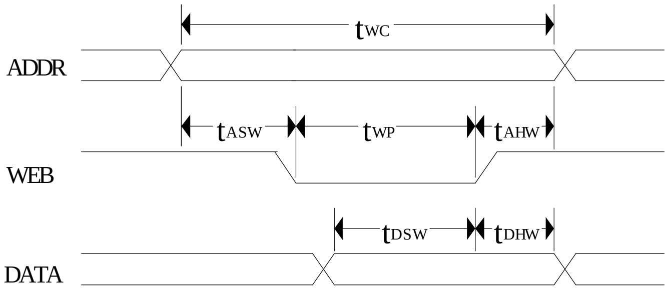


Figure 7.3.2 Timing of WIO write cycle

Symbol	Parameter	Min	Max	Unit
t_{wc}	Write cycle time	4		PCICLK
t_{asw}	ADDR to WEB setup time	1		PCICLK
t_{wp}	WEB pulse width	2		PCICLK
t_{ahw}	ADDR to WEB hold time	1		PCICLK
t_{dsw}	DATA to WEB setup time	2		PCICLK
t_{dhw}	DATA to WEB hold time	1		PCICLK

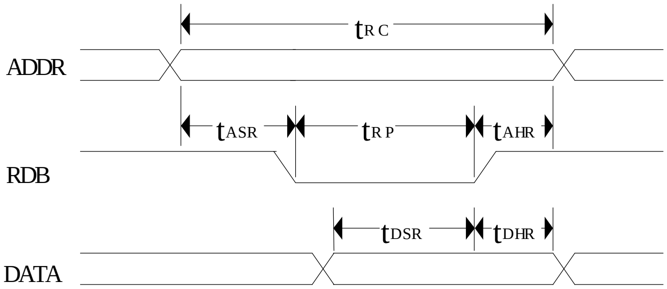


Figure 7.3.3 Timing of WIO read cycle

Symbol	Parameter	Min	Max	Unit
t_{rc}	Read cycle time	3		PCICLK
t_{asr}	ADDR to RDB setup time	1		PCICLK
t_{rp}	RDB pulse width	2		PCICLK
t_{ahr}	ADDR to RDB hold time	0		PCICLK
t_{dsr}	DATA to RDB setup time	2		PCICLK
t_{dhr}	DATA to RDB hold time	0		PCICLK

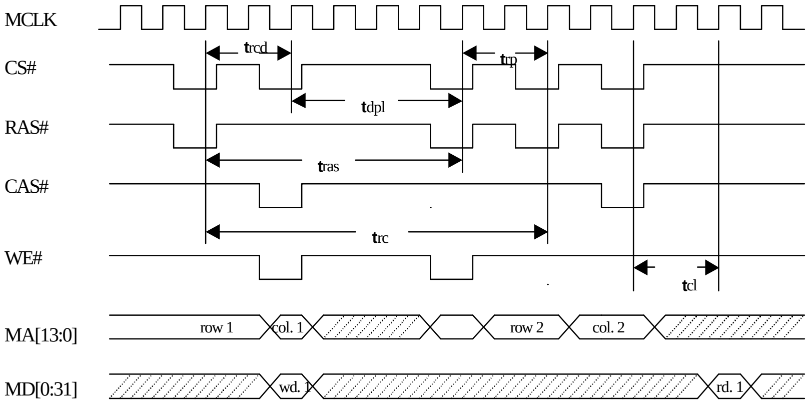


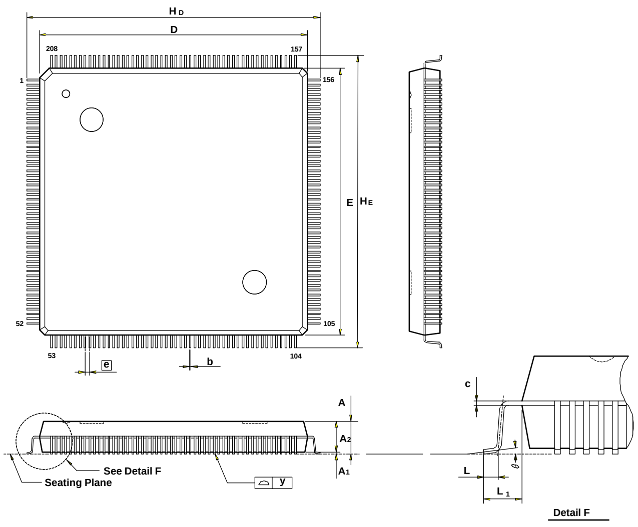
Figure 7.3.4 Timing of SDRAM

Symbol	Parameter	Min	Max	Unit
t_{rcd}	RAS# to CAS# delay	1	8	MCLK
t_{dpl}	Data-in to PRE Command Period	1	4	MCLK

t_{rp}	RAS# precharge time	1	8	MCLK
t_{ras}	RAS# active time	1	8	MCLK
t_{rc}	RAS# cycle time	2	9	MCLK
t_{cl}	CAS# latency time	1	3	MCLK
t_{casw}	CAS# pulse width for write	1	4	MCLK
t_{casr}	CAS# pulse width for read	1	8	MCLK
t_{cp}	CAS# precharge time	1	2	MCLK

8 PACKAGE DIMENSIONS

The W90221XX is packaged in a 208-pin PQFP package. The following figure shows its mechanical dimension



Symbol	Dimension in inch			Dimension in mm		
	Min	Nom	Max	Min	Nom	Max
A	—	—	0.145	—	—	3.68
A_1	0.004	—	—	0.10	—	—
A_2	0.122	0.127	0.132	3.10	3.23	3.35
b	0.006	0.008	0.010	0.15	0.20	0.25
c	0.004	0.006	0.010	0.10	0.15	0.25
D	1.097	1.102	1.107	27.87	28.00	28.13
E	1.097	1.102	1.107	27.87	28.00	28.13
e	0.016	0.020	0.024	0.40	0.50	0.60
H_D	1.193	1.205	1.217	30.30	30.60	30.90
H_E	1.193	1.205	1.217	30.30	30.60	30.90
L	0.012	0.020	0.028	0.30	0.50	0.70
L_1	0.043	0.051	0.059	1.10	1.30	1.50
y	—	—	0.004	—	—	0.10
θ	0°	—	10°	0°	—	10°

Note:

1. **Dimension D & E do not include interlead flash.**
2. **Dimension b does not include dambar protrusion/intrusion .**
3. **Controlling dimension : Millimeter**
4. **General appearance spec. should be based on final visual inspection spec.**

Figure 8.1 Package outline

APPENDIX A : ARCHITECTURE IMPLEMENT DEPENDENT REGISTERS

AIR[0] : Internal configuration register **(default : 12'b0)**

- bit 31 : Internal Icache enable (0/1 - disable/enable)
- bit 30 : Internal Dcache enable (0/1 - disable/enable)
- bit 29 : shall be the same as bit-30
- bit 28 : Default endian bit (0/1 - big-/little-endian)
- bit 27 : BTB enable (0/1 - disable/enable)
- bit 26 : reserved
- bit 25 : Multiplier fraction mode (0/1 - integer/fraction mode)
- bit 24 : reserved
- bit 23 : Freeze 1st 1K of Icache (0/1 - disable/enable)
- bit 22 : Freeze 2nd 1K of Icache (0/1 - disable/enable)
- bit 21 : Freeze 3rd 1K of Icache (0/1 - disable/enable)
- bit 20 : Freeze 4th 1K of Icache (0/1 - disable/enable)

AIR[1] : PSW register - for testing only

AIR[2] : TMR register - for testing only

AIR[3] : NonCacheable Offset register **(default : 32'b0)**

- bit 0-15 : reserve
- bit 16-19 : system non-cacheabel region
  0000 : all system memory are cacheable
  0001 : system memory above 1M are non-cacheable
  0010 : system memory above 2M are non-cacheable
  0011 : system memory above 4M are non-cacheable
  0100 : system memory above 8M are non-cacheable
  0101 : system memory above 16M are non-cacheable
  0110 : system memory above 32M are non-cacheable
  0111 : system memory above 64M are non-cacheable
  1000 : system memory above 128M are non-cacheable
  1001 : system memory above 256M are non-cacheable
- bit 20 : 0xA0000 ~ 0xFFFFF are non-cacheable
  0 : No
  1 : Yes
- bit 21-23: Size of non-cacheable region 1
  000 : disable
  001 : 64K
  001 : 128K
  001 : 256K
  001 : 512K
  001 : 1M
  001 : 2M
  001 : 4M

- bit 24-26: Size of non-cacheable region 2
  The bit definition is the same as that of bit 21-23. The base

address, defined by non-

cacheable base register (AIR[4]), will be on each regions' boundary automatically. (ie,
according to the size of each non-cacheable region, some LSBs of their relative base
register will be neglected.

- bit 27-29: reserved
- bit 30 : Data cache write-through mode
  0 : write-back data cache
  1 : write-through data cache
- bit 31 : reserved

AIR[4] : NonCacheable Base register **(default : 32'b0)**

- bit 0-15 : base address of non-cacheable region 1
- bit 16-31 : base address of non-cacheable region 2

AIR[5] : reserved

AIR[6] : reserved

AIR[7] : (HPSW[0:31])
Back up states of PSW as pipeline enter HALT state.

AIR[8] : (ICEA_Front, ICEA_Back)
Back up IAOQ_Frond and IAOQ_Back as pipeline enter HALT state

AIR[9] : (IDR[0:31]) ICE-Data register
This register is used for data exchange between ICE module and CPU.

AIR[10] : (ITR[0:31]) ICE-Trap register
This register defines which trap event would force pipeline enter HALT state.

AIR[11] : (PWR[0:2]) Power-Mode register

- bit 0 : (SLEEP) Force CPU into sleep mode
- bit 1 : (DOZE) Force CPU to doze-mode
- bit 2 : (STDBY) Force CPU to stand-by-mode

AIR[12] : (CKR[0:7]) Stand-By-Clock register
Define the clock rate (STDCLK) in Stand-By mode :
STDCLK = CPUCLK /((CKR[0:7]+1)*4)

APPENDIX B : DIAGNOSTIC EXTENDED INSTRUCTION SET

EXIT

Format: EXIT

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	-	-	-	-	0	10	-	-
6	5	5	2	2	1	5	1	5

Purpose: Returning instruction pipeline to normal mode from ICE mode.

Description: Instruction pipeline will be back to normal mode immediately after "EXIT" has being executed.

Operation:

```
PSW <-- HPSW;
IAOQ_Front <-- ICEA_Front;
IAOQ_Back <-- ICEA_Back;
```

Note : This instruction can be executed by code running at any privileged level, different from any other diagnostic instructions.

Move to AIR

Format: MTAIR r, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	t	r	-	-	0	00	-	-
6	5	5	2	2	1	5	1	5

Purpose : To copy value into a specified AIR from a general register.

Description : If the AIR[t] is existed, the contents of GR[r] is copied into AIR[t]. If AIR[t] has n bits where n <= 32, the least significant n bits of GR[r] are moved into AIR[t].

Operation :

```
if (t > 3)
        undefine operation;
else if (priv != 0)
        privilege instruction trap;
else
        AIR[t] <-- GR[r];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Note : AIR[0] : Internal configuration register **(default : 12'b0)**

- bit 31 : Internal Icache enable (0/1 - disable/enable)
- bit 30 : Internal Dcache enable (0/1 - disable/enable)
- bit 29 : shall be the same as bit-30
- bit 28 : Default endian bit (0/1 - big-/little-endian)
- bit 27 : BTB enable (0/1 - disable/enable)
- bit 26 : reserved
- bit 25 : Multiplier fraction mode (0/1 - integer/fraction mode)
- bit 24 : reserved
- bit 23 : Freeze 1st 1K of Icache (0/1 - disable/enable)
- bit 22 : Freeze 2nd 1K of Icache (0/1 - disable/enable)
- bit 21 : Freeze 3rd 1K of Icache (0/1 - disable/enable)
- bit 20 : Freeze 4th 1K of Icache (0/1 - disable/enable)

AIR[1]	: PSW register	**(default : 32'b0)**
AIR[2]	: TMR register	
AIR[3]	: NonCacheable Offset register	
AIR[4]	: NonCacheable Mask register	
AIR[5]	: Write-Through Offset register	
AIR[6]	: Write-Through Mask register	
AIR[7]	: HPSW[0:31]	
AIR[8]	: ICEA_Front, ICEA_Back	
AIR[9]	: IDR[0:31] (ICE-Data register)	
AIR[10]	: ITR[0:31] (ICE-Trap register)	
AIR[11]	: PWR[0:2] (Power-Mode register)	
AIR[12]	: CKR[0:7] (Stand-By-Clock register)	

Move from AIR

Format: MFAIR r, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r	-	-	-	0	01	-	t
6	5	5	2	2	1	5	1	5

Purpose : To copy value into a general register from AIR register.

Description : If the AIR[t] is existed, the contents of AIR[r] is copied into GR[t]. If AIR[r] has n bits
where n <= 32, the least significant n bits of AIR[r] are moved into GR[t].

Operation :

```
if (t > 3)
        undefine operation;
else if (priv != 0)
```

```
        privilege instruction trap;
else
        GR[t] <-- AIR[r];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Note : AIR[0] : Internal configuration register **(default : 12'b0)**
AIR[1] : PSW register **(default : 32'b0)**
AIR[2] : TMR register
AIR[3] : NonCacheable Offset register
AIR[4] : NonCacheable Mask register
AIR[5] : Write-Through Offset register
AIR[6] : Write-Through Mask register
AIR[7] : HPSW[0:31]
AIR[8] : ICEA_Front, ICEA_Back
AIR[9] : IDR[0:31] (ICE-Data register)
AIR[10] : ITR[0:31] (ICE-Trap register)
AIR[11] : PWR[0:2] (Power-Mode register)
AIR[12] : CKR[0:7] (Stand-By-Clock register)

Move to Btag

Format: MTBTAG b

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	04	-	-
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into BTB_tag from a general register.

Description : GR[b][0:23] is copied into a specified entry of BTB_tag.

Operation :

```
entry <-- GR[b][26:31];
set     <-- GR[b][24:25];

Btag[set, entry][0:26] <-- {GR[b][0:23], GR[b][21:23]};
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Note : Btag[set, entry][0:23] : tag field
Btag[set, entry][24] : valid bit
Btag[set, entry][25:26] : LRU bits

Move from Btag

Format: MFBTAG b,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	05	-	t
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into a general register from BTB_tag.

Description : A specified entry of BTB_tag is copied into GR[t].

Operation :

```
entry <-- GR[b][26:31];
set   <-- GR[b][24:25];

GR[t][0:26] <-- Btag[set, entry][0:26];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Move to Btable

Format: MTBTAB r,b

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	r	-	-	0	06	-	-
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into BTB_table from a general register.

Description : GR[r] is copied into a specified entry of BTB_table.

Operation :

```
entry <-- GR[b][26:31];
set   <-- GR[b][24:25];

Btable[set, entry][0:33] <-- {GR[r][0:31], GR[r][30:31]};
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Note : Btable[set, entry][0:31] : Branch target

Btable[set, entry][32:33]   : History bits

Move from Btable

Format: MFBTAB b,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	07	-	t
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into a general register from BTB_table.

Description : A specified entry of BTB_table is copied into GR[t].

Operation :

```
entry <-- GR[b][26:31];
set    <-- GR[b][24:25];
field  <-- GR[b][23];

if (field == 0)
        GR[t][0:31] <-- Btable[set, entry][0:31];
else
        GR[t][30:31] <-- Btable[set, entry][32:33];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Move to Itag

Format: MTITAG b

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	08	-	-
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into Itag from a general register.

Description : GR[b][0:20] is copied into a specified entry of BTB_tag.

Operation :

```
entry <-- GR[b][24:31];

Itag[entry][0:20] <-- GR[b][0:20];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Note : Itag[entry][0:19] : tag field
Itag[entry][20] : valid bit

Move from Itag

Format: MFITAG b,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	09	-	t
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into a general register from Itag.

Description : A specified entry of Itag is copied into GR[t].

Operation :

```
entry <-- GR[b][24:31];

GR[t][0:20] <-- Itag[entry][0:20];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Move to Icache

Format: MTICAH r,b

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	r	-	-	0	0A	-	-
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into Icache from a general register.

Description : GR[r] is copied into a specified entry of Icache.

Operation :

```
entry <-- GR[b][20:27];
word  <-- GR[b][28:29];

Btable[entry, word][0:31] <-- GR[r][0:31];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Move from Icache

Format: MFICAH b,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	0B	-	t
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into a general register from Icache.

Description : A word of specified Icache_entry is copied into GR[t].

Operation :

```
entry <-- GR[b][20:27];
word <-- GR[b][28:29];

          GR[t][0:31] <-- Icache[entry, word][0:31];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Move to Dtag

Format: MTDTAG b

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	0C	-	-
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into Dtag from a general register.

Description : GR[b][0:22] is copied into a specified entry of Dtag.

Operation :

```
entry <-- GR[b][22:27];
set    <-- GR[b][30:31];

Dtag[set, entry][0:22] <-- GR[b][0:22];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Note : Dtag[set, entry][0:21] : tag field
Dtag[set, entry][22] : valid bit

Move from Dtag

Format: MFDTAG b,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	0D	-	t
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into a general register from Dtag.

Description : A specified entry of Dtag is copied into GR[t].

Operation :

```
entry <-- GR[b][22:27];
set   <-- GR[b][30:31];

GR[t][0:22] <-- Dtag[set, entry][0:22];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Move to Dcache

Format: MTDCAH r,b

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	r	-	-	0	0E	-	-
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into Dcache from a general register.

Description : GR[r] is copied into a specified entry of Dcache.

Operation :

```
entry <-- GR[b][22:27];
word <-- GR[b][28:29];
set   <-- GR[b][30:31];

Dcache[set, entry, word][0:33] <-- {GR[r][0:31], GR[b][20:21]};
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

Note : Dcache[set, entry, word][0:31]: data word field
Dcache[set, entry][32] : dirty bit
Dcache[set, entry][33] : nru bit

Move from Dcache

Format: MFDCAH b,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	b	-	-	-	0	0F	-	t
6	5	5	2	2	1	5	1	5

Purpose : To copy a value into a general register from Dcache.

Description : A word of specified Dcache_entry is copied into GR[t].

Operation :

```
entry <-- GR[b][22:27];
word <-- GR[b][28:29];
set     <-- GR[b][30:31];
field  <-- GR[b][21];

if (field == 0)
        GR[t][0:31]   <-- Dcache[set, entry, word][0:31];
else
        GR[t][30:31] <-- Dcache[set, entry][32:33];
```

Exception : Privilege instruction trap.

Restriction : This instruction can only be executed by code running at the most privilege level.

APPENDIX C : MULTIPLIER EXTENDED INSTRUCTION SET

Halfword Multiply HMUL

Format: HMUL, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	01	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply corresponding 16-bit sign halfword of two general registers.

Description: The corresponding 16-bit sign halfwords of GR[r1] and GR[r2] are arithme-tically multiplied. The multiply result is placed in 32-bit LO accumulate register and GR[t] register. The bit in AIR[25] which indicates operating in integer or fraction mode determines the high-order or low-order 16 bits of GR[r1], GR[r2] will be as the two operands.

The completer, *cmplt*, determines multiplication in rounding or unrounding mode is performed, the completer specified by "r" indicates operating in rounding mode, the multiply result can be truncated the lower 16 bits when the least 16th bit is zero. IF the the least 16th bit is one, add one with the high-order 48 bits and truncate the low-order 16 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
    case r : (r=1, rounding mode){
        GR[t]{0:31} ← (sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31})+16h8000);
        LO{0:31} ← (sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31})+16h8000);
        break;
        }
    default : (r=0, unrounding mode){
        GR[t]{0:31} ← (sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31}));
        LO{0:31} ← (sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31}));
        break;
        }
}

Fraction mode operation (AIR[25] = 1):
switch(cmplt){
    case r : (r=1, rounding mode){
```

```
                GR[t]{0:31}        ←        (lshift(sign_ext(GR[r1]{0:15})        *
sign_ext(GR[r2]{0:15}), 1)
                +16h8000);
                LO{0:31}                ←    (lshift(sign_ext(GR[r1]{0:15})    *
sign_ext(GR[r2]{0:15}), 1)
                +16h8000);
                break;
                }
    default : (r=0, unrounding mode){
                GR[t]{0:31}        ←        lshift(sign_ext(GR[r1]{0:15})        *
sign_ext(GR[r2]{0:15}), 1);
                LO{0:31}                ←    lshift(sign_ext(GR[r1]{0:15})    *
sign_ext(GR[r2]{0:15}), 1);
                break;
                }
}
```

Exception : None

Halfword Multiply Unsign — HMULU

Format: HMULU, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	00	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply corresponding 16-bit unsign halfword of two general registers.

Description: The corresponding 16-bit unsign halfwords of GR[r1] and GR[r2] are arithme-tically multiplied. The multiply result is placed in 32-bit LO accumulate register and GR[t] register. The bit in AIR[25] which indicates operating in integer or fraction mode determines the high-order or low-order 16 bits of GR[r1], GR[r2] will be as the two operands.

The completer, *cmplt*, determines multiplication in rounding or unrounding mode is performed, the completer specified by "r" indicates operating in rounding mode, the multiply result can be truncated the lower 16 bits when the least 16th bit is zero. IF the the least 16th bit is one, add one with the high-order 48 bits and truncate the low-order 16 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
      case r : (r=1, rounding mode){
```

```
                    GR[t]{0:31}        ←        (zero_ext(GR[r1]{16:31})        *
zero_ext(GR[r2]{16:31})+16h8000);
                     LO{0:31}               ←     (zero_ext(GR[r1]{16:31})      *
zero_ext(GR[r2]{16:31})+16h8000);
                     break;
                }
     default : (r=0, unrounding mode){
                    GR[t]{0:31}        ←        (zero_ext(GR[r1]{16:31})        *
zero_ext(GR[r2]{16:31}));
                    LO{0:31}                 ←     (zero_ext(GR[r1]{16:31})     *
zero_ext(GR[r2]{16:31}));
                    break;
                    }
}
```

```
Fraction mode operation (AIR[25] = 1) :
switch (cmplt) {
       case r : (r=1, rounding mode){
                    GR[t]{0:31}       ←       (lshift(zero_ext(GR[r1]{0:15})        *
zero_ext(GR[r2]{0:15}), 1)
                    +16h8000);
                    LO{0:31}               ←     (lshift(zero_ext(GR[r1]{0:15})     *
zero_ext(GR[r2]{0:15}), 1)
                    +16h8000);
                    break;
                    }
     default : (r=0, unrounding mode){
                    GR[t]{0:31}       ←        lshift(zero_ext(GR[r1]{0:15})        *
zero_ext(GR[r2]{0:15}), 1);
                    LO{0:31}               ←      lshift(zero_ext(GR[r1]{0:15})     *
zero_ext(GR[r2]{0:15}), 1);
                    break;
                    }
}
```

Exception : None

Multiply MUL

Format: MUL, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	11	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply corresponding 32-bit sign words of two general registers.

Description: The corresponding 32-bit sign words of GR[r1] and GR[r2] are arithmetically multiplied. The multiply results are placed in 64-bit {HI, LO} accumulate register and word result is placed in GR[t]. The bit in AIR[25] indicates operating in integer or fraction mode.

The completer, *cmplt*, specified by "r" indicates operating in rounding mode, the multiply result can be truncated the lower 32 bits when the least 32th bit is zero. IF the the least 32th bit is one, add one the high-order 32 bits and truncate the low-order 32 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
    case r : (r=1, rounding mode){
            GR[t]{0:31} ← ((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2]), 64)+32h80000000){0:31};
            {HI, LO}    ← ((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2], 64)+32h80000000);
            break;
            }
   default : (r=0,  unrounding mode){
            GR[t]{0:31} ← ((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2]), 64)){32:63};
            {HI, LO}    ← ((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2]), 64));
            break;
            }
}

Fraction mode operation (AIR[25] = 1) :
switch(cmplt) {
    case r : (r=1, rounding mode){
            GR[t]{0:31} ← (lshift((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2]), 64), 1)
            +32h80000000){0:31};
            {HI, LO}    ← (lshift((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2]), 64), 1)
            +32h80000000);
            break;
            }
   default : (r=0,  unrounding mode){
            GR[t]{0:31} ← lshift((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2]), 64), 1){0:31};
            {HI, LO}    ← lshift((sign_ext(GR[r1]), 64) * (sign_ext(GR[r2]), 64), 1);
```

```
                break;
                }
}
```

Exception : None

Multiply Unsign MULU

Format: MULU, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	10	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply corresponding 32-bit sign words of two general registers.

Description: The corresponding 32-bit sign words of GR[r1] and GR[r2] are arithmetically multiplied. The multiply results are placed in 64-bit {HI, LO} accumulate register and word result is placed in GR t. The bit in AIR[25] indicates operating in integer or fraction mode.

The completer, *cmplt*, specified by "r" indicates operating in rounding mode, the multiply result can be truncated the lower 32 bits when the least 32th bit is zero. IF the the least 32th bit is one, add one the high-order 32 bits and truncate the low-order 32 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
      case r : (r=1, rounding mode){
                GR[t]{0:31} ← ((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]), 64))+32h80000000){0:31};
                {HI, LO}    ← ((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]), 64)+32h80000000;
                break;
                }
    default : (r=0, unrounding mode){
                GR[t]{0:31} ← ((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]), 64)){32:63};
                {HI, LO}  ← ((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]), 64));
                break;
                }
}

Fraction mode operation (AIR[25] = 1) :
switch (cmplt) {
        case r : (r=1, rounding mode){
                    GR[t]{0:31} ← (lshift((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]),
64), 1)
                    +32h80000000){0:31};
                     {HI, LO}      ← (lshift((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]),
64), 1)
                    +32h80000000;
                    break;
                    }
      default : (r=0, unrounding mode){
                    GR[t]{0:31} ← lshift((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]),
64), 1){0:31};
                    {HI, LO}       ← lshift((zero_ext(GR[r1]), 64) * (zero_ext(GR[r2]),
64), 1);
                    break;
                    }
}
```

Exception : None

Move To HI MTHI

Format: MTHI, r

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	~	r	~	~	1	06	~	~
6	5	5	3	1	1	5	1	5

Purpose: To move a general register into high-order 32-bit accumulator register HI .

Description: Load the contents of general-purpose registor GR[r] into 32 bits high-order accumulator register HI and automatic sign_extended into 4 hidden bits.

Operation :

```
                  HI  ← GR[r];
    HIDDEN[0:3]  ← {4{GR[r]{0}}};
```

*** HIDDEN : Hidden bits are implemented to avoid saturation in word operation. (reference SAT inatruction).**

Exception : None

Move To LO MTLO

Format: MTLO, cmplt, r

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	~	r	~	c	1	07	~	~
6	5	5	3	1	1	5	1	5

Purpose: To move a general register into low-order 32-bit accumulator register LO .

Description: Load the contents of general-purpose registor GR[r] into 32 bits low-order accumulator register. The *cmplt,* c determines whether automatic sign_extended to the 32 bits high-order accumulator HI;

Operation :

```
switch (cmplt){
        case e : (c = 1) {
                  LO              ← GR[r];
                  HI              ← {32{GR[r]{0}}};
                  HIDDEN[0:3]  ← {4{GR[r]{0}}};
                  }
        default : (c = 0) {
                  LO ← GR[r];
                  }
}
```

Exception : None

Move From HI MFHI

Format: MFHI, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	0	0	~	~	1	09	~	t
6	5	5	3	1	1	5	1	5

Purpose: To move the high-order 32-bits of the accumulator HI into a general register GR[t].

Description: The high-order 32 bits accumulator register HI is stored into a general register GR[t].

Operation :

```
        GR[t] ← HI[0:31];
```

Exception : None

Move From LO MFLO

Format: MFLO, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	0	0	~	~	1	08	~	t
6	5	5	3	1	1	5	1	5

Purpose: To move the low-order 32-bits accumulator register LO into a general register GR[t].

Description: The low-order 32 bits accumulator register LO is stored into a general register GR[t].

Operation :

GR[t] ← LO[0:31];

Exception : None

Halfword Multiply And Accumulate HMAC

Format: HMAC, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	03	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two signed 16-bit halfword of GR[r1] register and GR[r2] register, then accumulate {HI, LO} register with the multiplied result.

Description: The corresponding 16-bit halfwords of GR[r1] and GR[r2] are interpretted as signed 16-bit operands, and are arithmetically multiplied and add the product to the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register and GR[t], the bit in AIR[25] which indicates operating in integer or fraction mode determines the high-order halfword or low-order halfword of GR[r1], GR[r2] will be as the two operands .

The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 16 bits when the least 16th bit is zero. IF the the least 16th bit is one, add one the high-order 16 bits and truncate the low-order 16 bits.

Operation:

Integer mode operation (AIR[25] = 0) :

```
switch (cmplt) {
        case r: (r=1, rounding mode){
                  GR[t]{0:31}   ←   ({HI,   LO}   +   ((sign_ext(GR[r1]{16:31})   *
sign_ext(GR[r2]{16:31})
                                       + 16h8000)){32:63};
                  {HI,  LO}            ←  ({HI,  LO}  +  ((sign_ext(GR[r1]{16:31})  *
sign_ext(GR[r2]{16:31})
                                       +16h8000)){0:63};
                  break;
                  }
     default : (r=0, unrounding mode){
                 GR[t]{0:31}   ←   {HI,   LO}   +   sign_ext(GR[r1]{16:31})   *
sign_ext(GR[r2]{16:31}){32:63};
                {HI,  LO}                ←  {HI,  LO}  +  sign_ext(GR[r1]{16:31})  *
sign_ext(GR[r2]{16:31});
                 break;
                }
}
```

```
Fraction mode opeartion (AIR[25] = 1) :
switch(cmplt){
     case r: (r=1, rounding mode){
               GR[t]{0:31}  ←   ({HI,   LO}   +   (lshift(sign_ext(GR[r1]{0:15})   *
sign_ext(GR[r2]{0:15}),
                                      1) + 16h8000)){32:63};
               {HI,  LO}           ←  ({HI,  LO}  +  (lshift(sign_ext(GR[r1]{0:15})  *
sign_ext(GR[r2]{0:15}),
                                     1) + 16h8000)){0:63};
               break;
               }
   default : (r=0, unrounding mode){
               GR[t]{0:31}  ←   ({HI,   LO}   +   (lshift(sign_ext(GR[r1]{0:15})   *
sign_ext(GR[r2]{0:15}),
                                     1){32:63};
              {HI,  LO}            ←  ({HI,  LO}  +  (lshift(sign_ext(GR[r1]{0:15})  *
sign_ext(GR[r2]{0:15}),
                                     1){0:63};
               break;
                 }
}
```

Exception : None

Halfword Multiply And Accumulate Unsign HMACU

Format: HMACU, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	02	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two unsigned 16-bit halfword of GR[r1] and GR[r2] register, then accumulate {HI, LO} register with the multiplied result.

Description: The corresponding 16-bit halfwords of GR[r1] and GR[r2] are interpretted as unsigned 16-bit operands, and are arithmetically multiplied and add the product to the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register and GR[t], the bit in AIR[25] which indicates operating in integer or fraction mode determines the high-order halfword or low-order halfword of GR[r1], GR[r2] will be as the two operands .

The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 16 bits when the least 16th bit is zero. IF the the least 16th bit is one, add one the high-order 16 bits and truncate the low-order 16 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
    case r : (r=1, rounding mode){
        GR[t]{0:31} ← ({HI, LO} + ((zero_ext(GR[r1]{16:31}) * zero_ext(GR[r2]{16:31})
                    + 16h8000)){32:63};
        {HI, LO} ← ({HI, LO} + (zero_ext(GR[r1]{16:31}) * zero_ext(GR[r2]{16:31})
                    +16h8000)){0:63};
        break;
        }
    default : (r=0, unrounding mode){
        GR[t]{0:31} ← {HI, LO} + zero_ext(GR[r1]{16:31}) * zero_ext(GR[r2]{16:31}){32:63};
        {HI, LO} ← {HI, LO} + zero_ext(GR[r1]{16:31}) * zero_ext(GR[r2]{16:31});
        break;
        }
}

Fraction mode operation (AIR[25] = 1) :
switch(cmplt) {
    case r : (r=1, rounding mode){
        GR[t]{0:31} ← ({HI, LO} + lshift(zero_ext(GR[r1]{0:15}) * zero_ext(GR[r2]{0:15}),
                    1) + 16h8000){32:63};
```

```
                {HI,  LO}          ←  ({HI,  LO}  +  lshift(zero_ext(GR[r1]{0:15})  *
zero_ext(GR[r2]{0:15}),
                                       1) + 16h8000){0:63};
                 break;
                }
   default : (r=0, unrounding mode){
                GR[t]{0:31}   ←   ({HI,   LO}   +   lshift(zero_ext(GR[r1]{0:15})   *
zero_ext(GR[r2]{0:15}),
                                        1)){32:63};
                {HI,  LO}          ←  ({HI,  LO}  +  lshift(zero_ext(GR[r1]{0:15})  *
zero_ext(GR[r2]{0:15}),
                                        1)){0:63};
                 break;
                }
}
```

Exception : None

Halfword Multiply And Subtract HMSB

Format: HMSB, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	05	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two sign 16-bit halfword of GR[r1] and GR[r2], then subtract the multiplied result from {HI, LO} accumulate register.

Description: The corresponding 16-bit halfwords of GR[r1] and GR[r2] are interpretted as signed operands, and are arithmetically multiplied and subtract the product from the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register, and word result is placed in GR[t]. The bit in AIR[25] which indicates operates in integer or fraction mode determines the high-order halfword or low-order halfword of GR[r1], GR[r2] will be as the two operands .

The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 16 bits when the least 16th bit is zero. IF the the least 16th bit is one, add one the high-order 16 bits and truncate the low-order 16 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
       case r: (r=1, rounding mode){
            GR[t]{0:31} ← ({HI, LO} - ((sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31})
                        + 16h8000)){32:63};
            {HI, LO} ← ({HI, LO} - ((sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31})
                        +16h8000)){0:63};
            break;
            }
    default : (r=0, unrounding mode){
            GR[t]{0:31} ← {HI, LO} - sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31}){32:63};
            {HI, LO} ← {HI, LO} - sign_ext(GR[r1]{16:31}) * sign_ext(GR[r2]{16:31});
            break;
            }
}
```

Fraction mode opeartion (AIR[25] = 1) :

```
switch(cmplt){
    case r: (r=1, rounding mode){
            GR[t]{0:31} ← ({HI, LO} - (lshift(sign_ext(GR[r1]{0:15}) * sign_ext(GR[r2]{0:15}),
                        1) + 16h8000)){32:63};
            {HI, LO} ← ({HI, LO} - (lshift(sign_ext(GR[r1]{0:15}) * sign_ext(GR[r2]{0:15}),
                        1) + 16h8000)){0:63};
            break;
            }
  default : (r=0, unrounding mode){
            GR[t]{0:31} ← ({HI, LO} - (lshift(sign_ext(GR[r1]{0:15}) * sign_ext(GR[r2]{0:15}),
                        1){32:63};
            {HI, LO} ← ({HI, LO} - (lshift(sign_ext(GR[r1]{0:15}) * sign_ext(GR[r2]{0:15}),
                        1){0:63};
            break;
            }
}
```

Exception : None

Halfword Multiply And Subtract Unsign HMSBU

Format: HMSBU, cmplt r1,r2,t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	04	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two unsigned 16-bit halfword of GR[r1] and GR[r2] , then subtract the multiplied result from {HI, LO} accumulate register..

Description: The corresponding 16-bit halfwords of GR[r1] and GR[r2] are interpretted as unsigned 16-bit operands, and are arithmetically multiplied, then subtract the product from the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register and GR[t], the bit in AIR[25] which indicates operating in integer or fraction mode determines the high-order halfword or low-order halfword of GR[r1], GR[r2] will be as the two operands .

The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 16 bits when the least 16th bit is zero. IF the the least 16th bit is one, add one the high-order 16 bits and truncate the low-order 16 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
      case r : (r=1, rounding mode){
               GR[t]{0:31}  ←  ({HI,  LO}  -  ((zero_ext(GR[r1]{16:31})  *  zero_ext(GR[r2]{16:31})
                                    + 16h8000)){32:63};
               {HI,  LO}        ←  ({HI,  LO}  -  (zero_ext(GR[r1]{16:31})  *  zero_ext(GR[r2]{16:31})
                                    +16h8000)){0:63};
               break;
               }
    default : (r=0, unrounding mode){
              GR[t]{0:31}  ←  {HI,  LO}  -  zero_ext(GR[r1]{16:31})  *  zero_ext(GR[r2]{16:31}){32:63};
              {HI,  LO}          ←  {HI,  LO}  -  zero_ext(GR[r1]{16:31})  *  zero_ext(GR[r2]{16:31});
              break;
               }
}

Fraction mode operation (AIR[25] = 1) :
switch(cmplt) {
    case r : (r=1, rounding mode){
             GR[t]{0:31}  ←  ({HI,  LO}  -  lshift(zero_ext(GR[r1]{0:15})  *  zero_ext(GR[r2]{0:15}),
                                  1) + 16h8000){32:63};
             {HI,  LO}         ←  ({HI,  LO}  -  lshift(zero_ext(GR[r1]{0:15})  *  zero_ext(GR[r2]{0:15}),
                              1) + 16h8000){0:63};
              break;
             }
   default : (r=0, unrounding mode){
             GR[t]{0:31}  ←  ({HI,  LO}  -  lshift(zero_ext(GR[r1]{0:15})  *
zero_ext(GR[r2]{0:15}),
                              1)){32:63};
             {HI,  LO}        ←  ({HI,  LO}  -  lshift(zero_ext(GR[r1]{0:15})  *
zero_ext(GR[r2]{0:15}),
                              1)){0:63};
              break;
             }
}
```

Exception : None

Multiply And Accumulate MAC

Format: MAC, cmplt r1,r2,t

0:5	6:10	11:1	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	13	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two sign 32-bit word of GR[r1] and GR[r2], then add accumulate register {HI, LO} with the multiplied result.

Description: The corresponding 32-bit word of GR[r1] and GR[r2] are interpretted as signed 32-bit operands, and are arithmetically multiplied and add the product with the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register. The bit in AIR[25] indicates operating in integer or fraction mode.

The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 32 bits when the least 32th bit is zero. IF the the least 32th bit is one, add one the high-order 32 bits and truncate the low-order 32 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
       case r : (r=1, rounding mode){
                GR[t]{0:31}  ←  ({HI,  LO}  +  (sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31})
                                 + 32h80000000){0:31};
                {HI,  LO}        ←  ({HI,  LO}  +  (sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31})
```

```
                                        + 32h80000000){0:63};
                    break;
                   }
      default : (r=0, unrounding mode){
                   GR[t]{0:31}   ←   {HI,   LO}   +   sign_ext(GR[r1]{0:31})   *
sign_ext(GR[r2]{0:31}){32:63};
                   {HI,  LO}         ←  {HI,  LO}  +  sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31});
                    break;
                   }
}
```

```
Fraction mode operation (AIR[25] = 1) :
switch (cmplt) {
     case r : (r=1, rounding mode){
                  GR[t]{0:31}  ←  ({HI,   LO}  +  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                      1) + 32h80000000){0:31};
                  {HI,  LO}        ←  ({HI,  LO}  +  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                      1) + 32h80000000){0:63};
                   break;
                  }
    default : (r=0, urounding mode){
                   GR[t]{0:31}  ←  ({HI,   LO}  +  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                       1){0:31};
                   {HI,  LO}        ←  ({HI,  LO}  +  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                       1){0:63};
                    break;
                  }
}
```

Exception : None

Multiply And Accumulate Unsign
MACU

Format: MACU, cmplt r1,r2,t

0:5	6:10	11:1	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	12	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two unsign 32-bit word of GR[r1] and GR[r2], then add accumulate register {HI, LO} with the multiplied result.

Description: The corresponding 32-bit word of GR[r1] and GR[r2] are interpretted as unsigned 32-bit operands, and are arithmetically multiplied and add the product with the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register. The bit in AIR[25] indicates operating in integer or fraction mode.

The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 32 bits when the least 32th bit is zero. IF the the least 32th bit is one, add one the high-order 32 bits and truncate the low-order 32 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
        case r : (r=1, rounding mode){
                GR[t]{0:31} ← ({HI, LO} + (zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31})
                                    + 32h80000000)){0:31};
                {HI, LO} ← ({HI, LO} + (zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31})
                                    +32h80000000)){0:63};
                break;
               }
     default : (r=0, unrounding mode){
               GR[t]{0:31} ← {HI, LO} + zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}){32:63};
               {HI, LO} ← {HI, LO} + zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31});
               break;
              }
}

Fraction mode operation :
switch(cmplt) {
      case r : (r=1, rounding mode){
              GR[t]{0:31} ← ({HI, LO} + lshift(zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}),
                                  1) + 32h80000000){0:31};
              {HI, LO} ← ({HI, LO} + lshift(zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}),
                                  1) + 32h80000000){0:63};
               break;
             }
    default : (r=0, unrounding mode){
              GR[t]{0:31} ← ({HI, LO} + lshift(zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}),
                              1){0:31};
              {HI,  LO}         ←  ({HI,  LO}  +  lshift(zero_ext(GR[r1]{0:31})  *
zero_ext(GR[r2]{0:31}),
                              1){0:63};
              break;
             }
}
```

Exception : None

Multiply And Subtract MSB

Format: MSB, cmplt r1,r2,t

0:5	6:10	11:1	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	15	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two sign 32-bit word of GR[r1] and GR[r2], then subtract the product from accumulate register {HI, LO}.

Description: The corresponding 32-bit word of GR[r1] and GR[r2] are interpretted as signed 32-bit operands, and are arithmetically multiplied and subtract the product from the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register. The bit in AIR[25] indicates operating in integer or fraction mode.
The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 32 bits when the least 32th bit is zero. IF the the least 32th bit is one, add one the high-order 32 bits and truncate the low-order 32 bits.

Operation:

```
Integer mode operation (AIR[25] = 0) :
switch (cmplt) {
      case r : (r=1, rounding mode){
              GR[t]{0:31}   ←   ({HI,   LO}   -   (sign_ext(GR[r1]{0:31})   *
sign_ext(GR[r2]{0:31})
                                 + 32h80000000){0:31};
              {HI,  LO}          ←  ({HI,  LO}  -  (sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31})
                                 + 32h80000000){0:63};
               break;
              }
    default : (r=0, unrounding mode){
             GR[t]{0:31}   ←   {HI,   LO}   -   sign_ext(GR[r1]{0:31})   *
sign_ext(GR[r2]{0:31}){32:63};
```

```
                {HI,  LO}           ←  {HI,  LO}  -  sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31});
                 break;
                }
}
```

Fraction mode operation (AIR[25] = 1) :

```
switch (cmplt) {
    case r : (r=1, rounding mode){
              GR[t]{0:31}  ←  ({HI,  LO}  -  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                     1) + 32h80000000){0:31};
              {HI,  LO}        ←  ({HI,  LO}  -  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                     1) + 32h80000000){0:31};
               break;
              }
   default : (r=0, urounding mode){
               GR[t]{0:31}  ←  ({HI,  LO}  -  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                      1){0:31};
               {HI,  LO}       ←  ({HI,  LO}  -  lshift(sign_ext(GR[r1]{0:31})  *
sign_ext(GR[r2]{0:31}),
                                      1){0:63};
                break;
              }
}
```

Exception : None

Multiply And Subtract Unsign MSBU

Format: MSBU, cmplt r1,r2,t

0:5	6:10	11:1	16:18	19	20	21:25	26	27:31
05	r2	r1	~	r	1	04	~	t
6	5	5	3	1	1	5	1	5

Purpose: To multiply two unsign 32-bit word of GR[r1] and GR[r2], then add accumulate register {HI, LO} with the multiplied result.

Description: The corresponding 32-bit word of GR[r1] and GR[r2] are interpretted as unsigned 32-bit operands, and are arithmetically multiplied and add the product with the present contents of the {HI, LO} register, the 64-bit result is placed in {HI, LO} register. The bit in AIR[25] indicates operating in integer or fraction mode.

The completer "r" indicates operating in rounding mode, the multiply result can be truncated the lower 32 bits when the least 32th bit is zero. IF the the least 32th bit is one, add one the high-order 32 bits and truncate the low-order 32 bits.

Operation:

Integer mode operation (AIR[25] = 0) :

```
switch (cmplt) {
    case r : (r=1, rounding mode){
        GR[t]{0:31} ← ({HI, LO} - (zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31})
                          + 32h80000000)){0:31};
        {HI, LO} ← ({HI, LO} - (zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31})
                          +32h80000000)){0:63};
        break;
        }
    default : (r=0, unrounding mode){
        GR[t]{0:31} ← {HI, LO} - zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}){32:63};
        {HI, LO} ← {HI, LO} - zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31});
        break;
        }
}
```

Fraction mode operation :

```
switch(cmplt) {
    case r : (r=1, rounding mode){
        GR[t]{0:31} ← ({HI, LO} - lshift(zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}),
                          1) + 32h80000000){0:31};
        {HI, LO} ← ({HI, LO} - lshift(zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}),
                          1) + 32h80000000){0:63};
        break;
        }
    default : (r=0, unrounding mode){
        GR[t]{0:31} ← ({HI, LO} - lshift(zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}),
                          1){0:31};
        {HI, LO} ← ({HI, LO} - lshift(zero_ext(GR[r1]{0:31}) * zero_ext(GR[r2]{0:31}),
                          1){0:63};
        break;
        }
}
```

Exception : None

Saturation SAT

Format: SAT, cmplt, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	r2	r1	~	c	1	16	~	t
6	5	5	3	1	1	5	1	5

Purpose: To test accumulate register {HI, LO} overflow, then saturate the accumulate register {HI, LO}

Description: To test accumulate register {HI, LO} overflow, and if the accumulate register {HI, LO} is overflow, then saturate the accumulate register, the accumulate register is not overflow, then perform a null operation.

The saturation instruction is intended to be used at the completion of a series of multiply/accumulate operations, so that temporary overflows do **NOT** occur the accumulator to saturate.

The saturation condition and result depend on operation type of halfword or word multiply/accumulate. If half word operation, test the high-order 33-bits of the accumulate register. The possible results after execution of saturation instruction is shown in Table I. If word operation, test the carry bit (hidden 4-bits) and sign bit, the possible results after execution of saturation instruction is shown in Table II.

The *cmplt* completer c specifies the saturation mode on halfword or word operation

Table I. (Halfword saturation operation mode).

*HMV	HI[0]		
0	0	No change	
0	1	No change	
1	0	64	0000_0000_7FFF_FFFF
1	1	64	FFFF_FFFF_8000_0000

*HMV : HMV is set when the HI[0:31] and sign-bit LO[0] are not all the same.

Table II. (Word saturation operation mode).

*WMV	*HIDDEN[0]		
0	0	No change	
0	1	No change	
1	0	64	7FFF_FFFF_FFFF_FFFF
1	1	64	8000 _0000 _0000 _0000

*WMV : WMV is set when the hidden bits and sign-bit HI[0] are not all the same.

*HIDDEN[0] : The most significiant bit of hidden bits.

Operation:

```
switch (cmplt) {
        case h : (c = 1; word saturation){
                if (HMV) {HI, LO}← {{33{HI[0]}}, {31{~HI[0]}}};
                        GR[t] ← {HI[0], {31{~HI[0]}}};
                else
                        GR[t] ← LO;
                break;
        }
        default : (c = 0; double word saturation){
                if (WMV) {HI, LO} ← {HIDDEN[0], {63{~HIDDEN[0]}}};
                        GR[t] ← {HIDDEN[0], {31{~HIDDEN[0]}}};
                else
                        GR[t] ← HI;
                break;
        }
}
```

Exception : None

Saturation GR[r] SATGR

Format: SATGR r, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	~	r	~	~	1	0c	~	t
6	5	5	3	1	1	5	1	5

Purpose: To test general register GR[r] 16-bit overflow, then saturate GR[r] into G[t],
GR r and Gr t may be the same register

Description: To test general GR[r] 16-bit overflow, and if the general register GR[r] is overflow, then saturate the general register GR[r], and the saturation result is put into destination register Gr[t]. If Gr[r] is not overflow, the GR[r] value is copied into GR[t].

The saturation instruction for GR[r] is intended to be used in the 16-bit halfword operation, and the result is stored in GR r register. The operation tests the high-order 32-bits of the register GR[r], and the possible results after execution of saturation is shown in TableI.

Taable I..
*HOV GR[r]{0}

0	0	No change
0	1	No change
1	0	32h0000_7FFF
1	1	32hFFFF_8000

*HOV : HOV is set when the high-order 17-bits of GR[r] register are not all the same.
Operation:

if(HOV) GR[t]{0:31} ← {{17{GR[r]{0}}}, {15{~GR[r]{0}}}};
break;

Exception : None

* Note : 16-bit halfword operation in PA-Architecture 32-bit operation. The MSB 16-bits should be sign_extented previously.

Derive Exponent EXP

Format: EXP, r, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	~	r	~	~	1	0d	~	t
6	5	5	3	1	1	5	1	5

Purpose: To derive the effective exponent of the operand GR[r] into GR[t].

Description: The EXP operation derives the effective exponent of the input operand to prepare for normalization. Generally speaking, Normalization can be divided into two-stage. The first stage derives the effective exponent. The second stage does the actually shifting. The first uses EXP instruction which detects the exponent value and load it into GR[t]. The second stage uses the shift relative instruction to shift GR[r] by the shift amount of GR[t].

Operation:

GR[t] ← Leading sign bits of GR[r].

Exception : None

Block Exponent Adjust ADJEXP

Format: ADJEXP, r, t

0:5	6:10	11:15	16:18	19	20	21:25	26	27:31
05	t	r	~	~	1	0e	~	t

6	5	5	3	1	1	5	1	5

Purpose: To perform on a series of numbers, derive the effective exponent of the number largest in magnitude.

Description: This function detects the effective exponent value of the number largest in magnitude in an array numbers. GR[t] remains the largest in magnitude in the effective exponent value of all previous numbers. Then compare exponent in magnitude of GR[r] with GR[t], if the effective exponent of current value (GR[r]) is smaller than the largest magnitude (GR[t]) in all previous numbers, update the GR[t] with the effective exponent value of GR[r]. Otherwise, remain the GR[t].

Operation:

```
if (EXP GR[r] < GR[t])
    GR[t] ← EXP GR[r];
```

EXC