PC202 Datasheet

PC202

Features

- picoArray Baseband Processor Array
- ARM®926EJ-S Processor
  - 280MHz
  - 64KB I + 64KB D cache
  - 128KB Tightly Coupled Memory (TCM)
  - 128KB on-chip SRAM
  - RS-232/JTAG for debug
- Quad FFT Block
- Dual Correlation Block
- Cryptographic Engine
  - AES, DES and 3DES
- FEC Accelerator
  - Turbo-code
  - Reed-Solomon
  - Viterbi
- 10/100 Ethernet MAC and MII
  - Reverse MII mode
- DDR2 SDRAM Interface
  - 16/32-bit
  - 200MHz clock
  - 256Mbit to 2Gbit
- Multiple Boot Options
  - Flash Memory
  - External Processor
  - MII Boot Packet
- General Purpose I/O - 32 pins
  - SIM Interface
  - Sigma Delta DAC
  - SPI/I^2C
- Commercial Temperature Range
- 672 PBGA Package (27x27)
  - Lead-free RoHS Compliant

Description

The picoChip PC202 is a highly integrated and cost effective baseband processor for broadband wireless access subscriber station applications.

The PC202 consists of a flexible software defined modem, powerful ARM®926EJ-S processor, cryptographic engine, optimized co-processors, and peripherals capable of supporting all of the mandatory features of WCDMA together with optional features, including advanced CTC (Convolutional Turbo Coder). All physical layer (PHY), lower MAC, upper MAC, and cryptographic features are integrated, enabling a greatly reduced BOM.

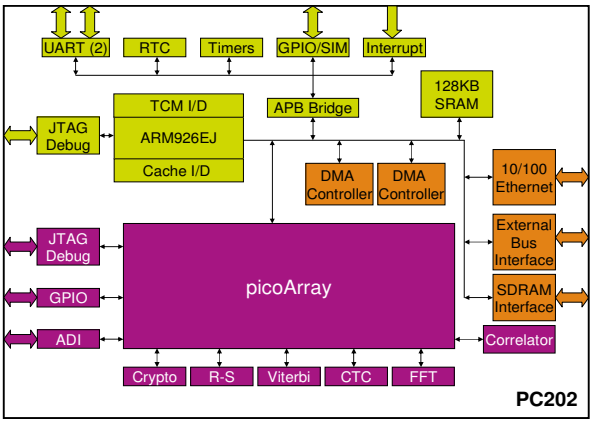
Flexible interfaces support multiple boot options, SDRAM memory, LAN, and radio interfaces minimize glue logic for a complete system solution.

A corresponding reference design for 802.16 PHY and MAC is available, enabling high-performance systems (improved rate/reach), beyond minimum performance requirements. The system supports upgrades (future-proof) to support interoperability, changes to the standard, or manufacturer specific features.

Applications

- HSDPA/HSUPA WCDMA Femtocell/Access Point
- CDMA 2000/EVDO Femtocells
- GSM/EDGE Femtocell
- Broadband Wireless Access
- Advanced Wireless
  - JTRS
  - Software Defined Radio
- Test and Measurement
- Medical Imaging

PC202 Simplified Block Diagram



PicoArray Sub-System

The picoArray is a software defined signal processor responsible for executing the PHY and lower MAC software. The advantage of picoArray is that it maintains flexibility within the PHY and lower MAC functions, allowing software upgrades to be simply applied as necessary in line with product release, localization, standardization updates, bug fixes, performance enhancements etc.

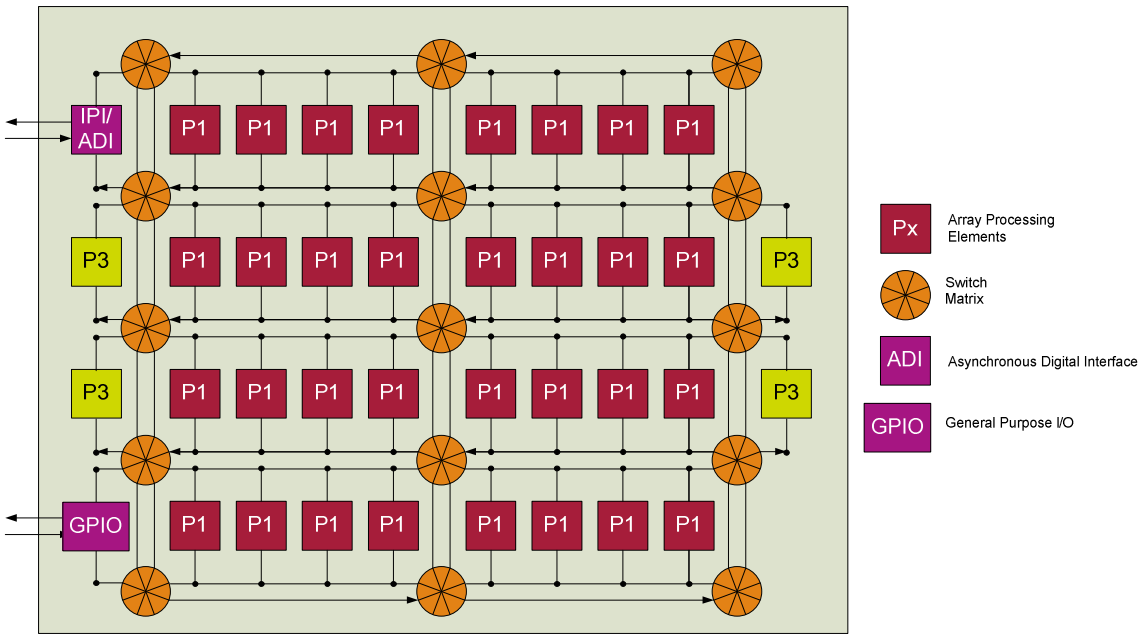
Multicore DSP Array (picoArray)

The picoArray consists an array of LIW DSPs (AEs) capable of up to 6 operations per cycle. Each AE has dedicated instruction memory and data memory. Each AE also has access to shared on-chip SRAM and off-chip DRAM. Dedicated instruction and data memory for each processor means very little contention for shared memory resources. There are three different types of AEs in the PC202. The table on the right lists the number and type of each AE along with its dedicated memory sizes.

AE Type	Number of AEs	Memory (Bytes)
STAN	196	768
MEM	50	8,704
CTRL	2	65,536
Total	248	716,800

Flexible interconnect provides point-to-point and point-to-multipoint connections between any two AEs with dedicated bandwidth.

Figure 1 picoArray of DSP Processors and Interconnect



Convolutional Turbo Code (CTC) Block

PC202 incorporates a hardware defined CTC block capable of supporting data rates 8 to 60 Mbps. The CTC algorithm which is optionally required in the SOFDMA air interface of IEEE 802.16e, and mandatory in TTA WiBRO allows much enhanced bit error rates in a data channel compared to CC and R-S coding. The CTC also supports all the 3GPP turbo code requirements. Soft input values are 8 bits, with a maximum number of 4800 soft inputs.

FFT/IFFT

FFT hardware accelerator designed to optimise the performance of OFDM based systems. The FFT engine can support complex FFT sizes of 128, 256, 512 and 1024 points. (FFTs up to 2048 points can be accommodated in software.) Features of the FFT block include on-the-fly FFT/IFFT, 16-bit input and output with scaling, bit reversal capability, and self-flushing mechanism.

Viterbi

Viterbi hardware accelerator block is used to accelerate the Viterbi decoding process in wireless systems. The Viterbi block is optimised for WiMAX and HSDPA systems. It is configurable for constraint lengths of 2 to 9, and rates of 1/2, 1/3, 1/4, 1/5, 1/6, 1/7 or 1/8. Multiple puncturing rates are also supported. Block sizes are from 16 to 1024.

Reed-Solomon (R-S)

The PC202 incorporates a Reed-Solomon accelerator block for use in R-S block coding requirements.

Feature	Perf.	Units
picoArray	31	GMAC/s
	230	GIP/s
	8	GMUL/s
FFT (aggregate)	80	MS/s
Crypto Engine	12	Mb/s
Turbo Code (8 iter.)	8	Mb/s
Viterbi	8	Mb/s
R-S	8	Mb/s
Correlator	40	Mb/s
ADI	1x160	MS/s
ARM	280	MHz

Cryptographic Engine

The PC202 incorporates the support of AES, DES & 3DES algorithms as required by the MAC security sub-layer and IPSec. Additional standards supported include NIST FIPS PUB 197, PUB 46-3, 800-38C, SP 800-38A.

Correlator Block

The PC202 contains two identical correlation accelerators. Each accelerator is a complex code matched filter (CCMF) for CDMA timing and acquisition (e.g. RACH preamble detection) schemes plus multi-path searching. It is a general purpose correlation engine for third generation CDMA based systems e.g. UMTS FDD, CDMA 2000 and TD-SCDMA. The blocks can perform up to 20 Billion correlations per second.

ARM® Sub-System

The ARM® sub-system is based around the ARM®926EJ-S core from ARM® Ltd. This powerful 32-bit RISC processor is highly optimized for low power, high performance computing. Being an industry standard core, the ARM® is supported by various openly available tool chains and debuggers. JTAG ports support a simple and familiar development environment.

ARM® Memory Architecture

The memory architecture supports a full range of memory types, allowing flexible memory map allocation depending upon the particular application requirements. The cache and tightly coupled memory deliver maximum processor performance. The memory types are summarised as follows

- On-chip memory for maximum performance
  - 64KB instruction, 64KB data cache, 32 bits wide
  - 64KB instruction, 64KB data TCM (Tightly Coupled Memory), 32-bit wide single state access
  - 128MB SRAM, 32 bits wide, two state access
- External memory
  - DDR2-SDRAM via dedicated interface
  - NOR Flash for system boot and parameter storage

ARM® Peripherals

PC202 integrates a number of peripheral blocks allowing a reduction in total system BOM cost, as well as providing flexible interfaces to application-specific sub-systems. The ARM® peripherals are logically mapped into the ARM® memory map, and are conveniently configurable by way of dedicated registers per block. The principle ARM® peripheral blocs are summarised as follows:

- General purpose timer block e.g. for operating system tick
- Watchdog Timer
- Dual UART RS-232 interface for real-time tracing of MAC software
- 10/100 Ethernet MAC
  - Reverse MII interface allowing direct connection to MII interfaces of router/WiFi chipsets
  - Built-in DMA controller
- GPIO – SPI radio control, I2C, SIM card interface
- Vectored Interrupt Controller

Applications Examples

WCDMA Home Basestation

An emerging application is the use of in-home internet access connections like ADSL and low-cost flexible wireless PHY technology like the PC202 to create an in-home 3G/2G femto-cell basestation. The PC202 PHY implementation and an ADSL link for backhaul allows mobile phone operators to offer a new level of service where customers can use their mobile phones at home; even in areas of poor coverage. picoChip offers a complete WCDMA reference design for small basestation applications.

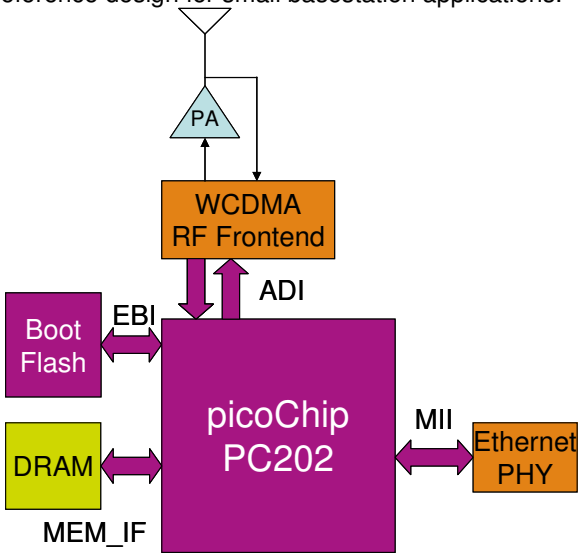


Figure 2 WCDMA Femtocell

Table of Contents

1 Scope

This Electrical Datasheet provides hardware engineers with detailed information for systems using the PC202 family of multi-core DSP chips. The picoArray supports high-speed digital interfaces for interconnecting with other picoArray devices and external hardware.

This Document focuses on the AC/DC characteristics, timing parameters, timing diagrams, ball out list, package diagram, pin out functionality, and the recommended handling of NC/Reserved pins.

1.1 What information is available

- For information on programming the picoArray, please refer to the picoTools documentation.
- The PC20X programmers guide describes the industry standard ARM926EJ sub-system and peripherals.
- Further functional Descriptions are available, please refer to the picoTools user Manual.

For further information and assistance, please contact the picoChip applications team at support@picochip.com.

2 Recommended Operating Conditions

Table 1: Recommended operating conditions

Symbol	Parameter	Conditions	Min	Typ	Max	Units
vddcore	Core Supply voltage		0.95	1	1.05	**V**
vddpll	PLL Supply Voltage		0.95	1	1.05	**V**
Tj	Junction temperature (T_J)		-40		100	**°C**
THjc	Thermal resistance junction to case (θ_{jc})			3.3		**°C per Watt**
THja	Thermal resistance junction to ambient (still air, no heatsink) (θ_{ja}) Note 2			12.5		**°C per Watt**
THjb	Thermal resistance junction to PCB (θ_{jb})			5.2		
	Storage temperature		-55		125	**°C**
Fclk	Input clock Frequency Fclk Note 1 Note 4			20		**MHz**
	Input clock duty cycle		40:60	50:50	60:40	**%**
Tjit	Input clock jitter (peak to peak) Note 5				+/-200	ps
Ivddcore	vddcore (1.0v) Note 3	Core				A
Ivddadi	vddadi (1.8v) Note 3	ADI, DDRII				A
Ivddio	vddio (3.3v) Note 3	EBI, MII, GPIO				A

Note 1: The input clock frequency Fclk is multiplied by the integrated PLL function to create the internal System Clocks required for the ARM Sub System and picoArray.

Note 2: Quoted figures assume 0 m/s airflow.

Note 3: For target applications, preliminary estimates of device dissipation at 90% picoArray core utilization and with input clock applied. Highest that might be reasonably expected for target applications, based on empirical data. Some cases may exceed this value.

Note 4: Crystal tolerance better than +/- 500ppm

Note 5: When driving xtal_in from external oscillator

Table 2: Device Power Consumption Analysis

Pin category	Comment	Conditions	Power 1.8V (mW)	Power 3.3V (mW)	Total Current (mA)
ADI	ADI port power	Vddadi(max) at 160MHz, 10pF	6.8		3.2
MEM_IF	DDR2 SDRAM Interface	vddmem(max) at 200MHz DDR terminated, 10pF	99		52
EBI/PROCIF	Flash/Processor Interface	vddio(max) at 100MHz, 50pF		319	92
MII	Ethernet MII Interface	vddio(max) at 25MHz, 8pF		3.8	1.08
GPIO	General Purpose I/O	vddio(max) at 20MHz, 20 pF		12	3.5
UART	Serial Ports	vddio(max) at 10MHz, 10 pF		0.9	0.26
MISC	JTAG	vddio(max) at 1MHz, 10 pF		0.13	0.04
Sync Out	Synch Signals	vddio(max) at 160MHz, 10 pF		26	7.6

Power Mode	Comment	Conditions	Power 1.0V (mW)	DC current (mA)
VDDcore static	Static power for VDDcore	vddcore(max), 100C junction temperature	1000	952
VDDcore dynamic	Dynamic power benchmark application	vddcore(max)	1936	1844
VDDcore total	Core total power static and dynamic	vddcore(max), 100C junction temperature	2936	2796
VDDcore reset	Total power when reset asserted	vddcore(max), 100C junction temperature	2730	2600

2.1 Electrical Characteristics

2.1.1 DC Characteristics

Table 3: MEM SSTL18 Interface DC Characteristics

Parameter	Conditions	Min	Typ	Max	Units
vddmem(vddq)		1.7	1.8	1.9	Volts
mem_vref (vref)		0.833	0.9	0.969	Volts
vtt		vref – 0.04	Vref	vref + 0.04	Volts
vih(dc)		vref + 0.125		vddq + 0.3	Volts
vil(dc)		-0.3		vref -0.125	Volts
Ioh(dc)	@Voh(dc)=vddq – 0.28v	-13.4			mA
Iol(dc)	@Vol(dc)= 0.28v	13.4			mA
Rt					Ohms
Rs					Ohms

Note: The PC202 data bus and data strobes have on die termination implemented.

Table 4: ADI DC Characteristics

Symbol	Description	Conditions	Min	Typ	Max	Units
vddadi	1.8v I/O supply voltage		1.7	1.8	1.9	Volts
voh(dc)	Output high voltage		vddadi – 0.4			Volts
vol(dc)	Output low voltage				0.4	Volts
Ioh(dc)	Output high current	@voh=vohmin	12			mA
Iol(dc)	Output low current	@vol=volmax	12			mA
vih	Input high threshold		0.65 vddio		5.5	Volts
vil	Input low threshold		-0.3		0.8	Volts
Ii	Input Leakage				+-10	uA
cio	Pin capacitance			tbd		pF

Table 5: 3.3v IO Interface DC Characteristics

Symbol	**Description**	**Conditions**	**Min**	**Typ**	**Max**	**Units**
vddio	3.3v I/O supply voltage		3.135	3.3	3.465	Volts
voh(dc)	Output high voltage		vddio – 0.4			Volts
vol(dc)	Output low voltage				0.4	Volts
Ioh(dc)	Output high current (MII/RS-232)	@voh=vohmin	8			mA
Iol(dc)	Output low current (MII/RS-232)	@vol=volmax	8			mA
Ioh(dc)	Output high current (GPIO/EBI)	@voh=vohmin	24			mA
Iol(dc)	Output low current (GPIO/EBI)	@vol=volmax	24			mA
vih	Input high threshold		0.65 vddio		5.5	Volts
vil	Input low threshold		-0.3		0.8	Volts
Ii	Input Leakage				+-10	uA
cio	Pin capacitance			tbd		pF

2.1.2 AC specifications

Table 6: SSTL18 AC Characteristics

Parameter	Conditions	Min	Typ	Max	Units
vindiff	vin-Vref		0.25		Volts
vih(ac)	vref + Vindiff	vref+0.25			Volts
vil(ac)	vref – Vindiff			vref-0.25	Volts
voh(ac)	Ioh = -13.4mA	vttmax+0.603			Volts
vol(ac)	Iol = 13.4mA			vttmax-0.603	Volts
vswing	peak-to-peak			1	Volts
slew		1			V/ns

2.2 Absolute Maximum Ratings

Table 7: Absolute Maximum Ratings

Parameter	Min	Typ	Max	Units
vddadi		Tbd		Volts
vddio		Tbd		Volts
vddcore		Tbd		Volts
Storage temperature range		Tbd		℃

Note: Any exposure to values greater than those given may cause permanent damage.

3 Boot Configuration

There are two main boot modes: master and slave. Master mode is intended for standalone systems or where the ARM926EJ is the processor responsible for configuring the picoArray™ and possibly a number of slave devices. In Master mode, it will have use of the EBI (Extension Bus Interface).

The EBI is a Master interface that can not be communicated to directly, it only has the ability to initiate accesses. The EBI will typically read boot information from external Flash memory.

In Slave mode, The devices will be part of a larger signal processing sub-system such as in macro base-station. One or more picoArray device will be connected to a master processor that is responsible for configuration, such as another PC202 or Microprocessor. E.G a PowerPC. Communication with the picoArray will be over the processor interface.

Table 8: Handling of Boot Mode Pins

boot_mode[1:0]	Mode
00	Master Mode EBI
01	Slave Mode Proc_IF

NOTE: boot_mode [1:0] pins must be static prior to /rst being released.

Figure 3: EBI Boot

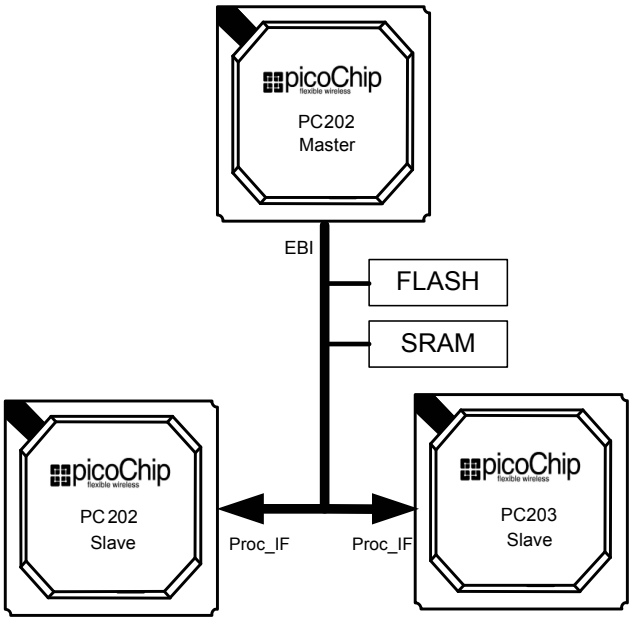


Figure 4: Stand alone Master Boot & Slave proc_IF Boot

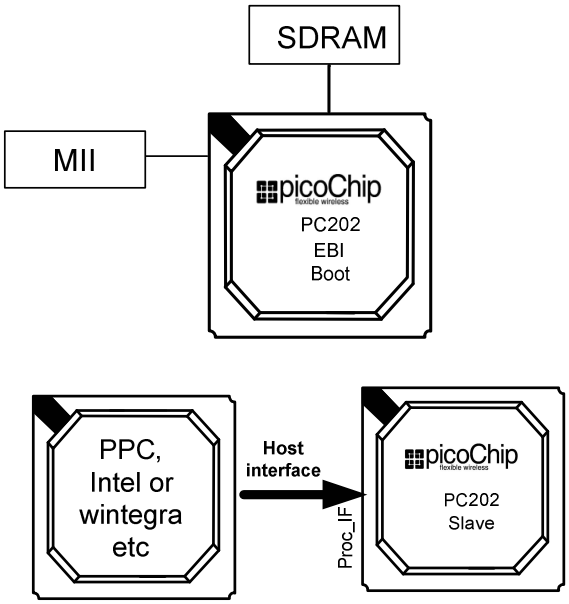


Figure 3 shows a Multiple picoArray system. The PC202 Boots from Flash the ARM926EJ coprocessor then boots subsequent PC20X Devices over the EBI to proc_IF interface

Figure 4 shows the PC202 as a stand alone device or Booted via a HOST processor.

3.1 Booting Via EBI and Proc_IF

Please refer to the PC20x ARM sub system programmer's guide and picoTools Manual for the description of Master/Slave boot process and memory map.

4 Timing parameters and diagrams

4.1 Clock Source

The PC202 has a built in crystal oscillator for generation of all internal clocks. An on-chip PLL is used to create the different internal clocks from the crystal oscillator clock. This includes the ARM9EJ processor, the picoArray, and the DRAM interface. The oscillator is connected as shown below. The crystal is 20MHz operating in fundamental mode.

Optionally the xtal_in pin signal can be driven by a CMOS clock signal at the specified frequency. In this case xtal_out is left unconnected.

Figure 5: Oscillator Connection

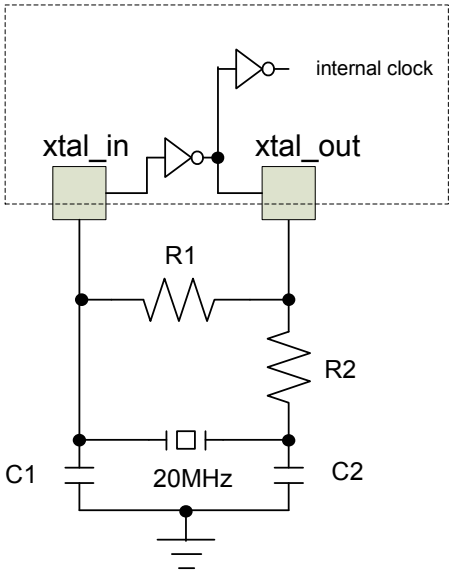


Table 9 Oscillator Operating Conditions

Description	Conditions	Min	Typ	Max	Units
20 MHz Crystal	NA	tbd		20	MHz
Series resistance	In	40		80	Ohms
R1	Rfb		1M		Ohms
R2	Rs		0		Ohms
Load Capacitance	(C1*C2)/(C1+ C2) including parasitic trace capacitance	8		20	pF
xtal_in Capacitance			15		pF
xtal_out Capacitance			15		pF
Input high threshold				1.47	Volts
Input Low threshold		1.28			Volts
Input Leakage				+-10	uA

4.1.1 Power-Up & Reset

The timing diagram, Figure 7 and Table 10 illustrate the power-up and reset sequence for the PC202 device:

Steps should be taken to avoid bus contention at power-up. This is achieved by applying reset to the device immediately the I/O (3.3V &1.8V) power is applied. All output pads float until core (1.0V) power is applied. With I/O (3.3 &1.8V) power applied and the /rst and /trst pins held low, apply core (1.0V) power. If core power is applied before reset and /trst are taken low then outputs will drive unknown data.

With core (1.0V) power applied, apply the input clock, (xtal_in) to the device. The reset pin must be held low for a minimum period of T4 after the crystal and core power have been applied. It forces the device bi-directional buses to float, resets all registers to a known, idle state and disables the picoArray core. This completes the device power-up and reset procedure

The core +1.0V supply must come up with, or after, the +1.8v and 3.3v I/O supplies. The simplest method is to connect a Schottky diode between the +1.0V, +1.8v and +3.3V supplies as shown in Figure 6.

Figure 6: Schottky diode between 3.3V, 1.8v and 1.0v supplies

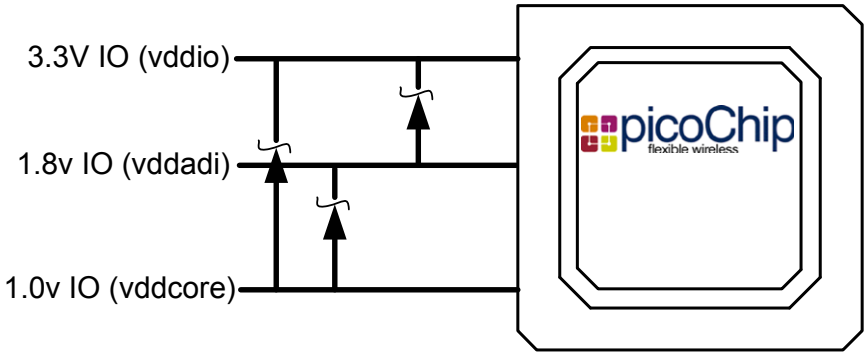


Figure 7: Power up Timing Diagram

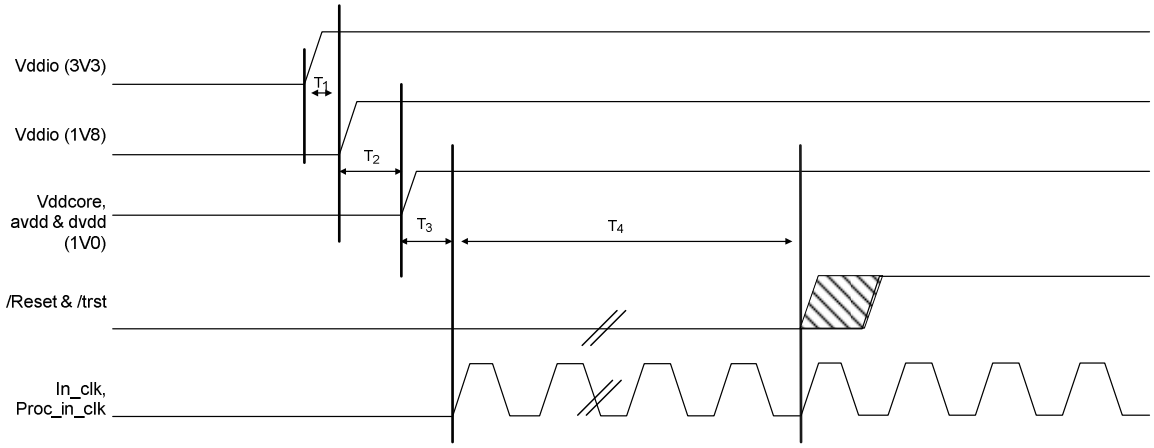


Figure 8: Reset Timing Diagram

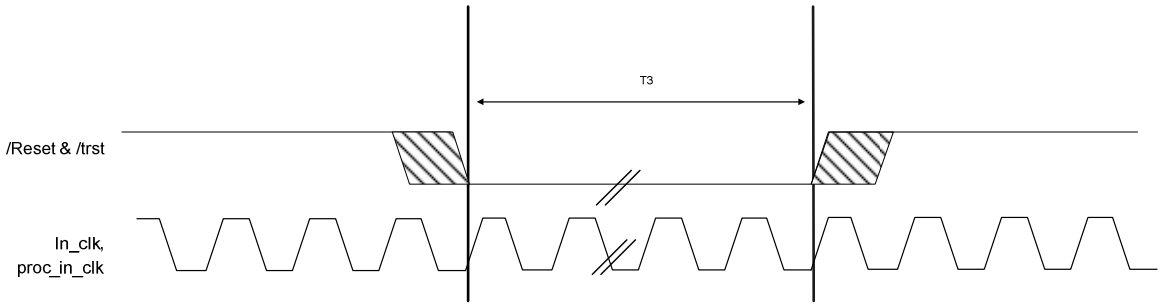


Table 10: PC202 Power up & Reset timings

Parameter	Description	Min	Max	Units
T_1	Delay Between Applying the 3.3V IO and 1.8V IO	0		ns
T_2	Delay between applying the 1.8V IO and 1.0V Core supplies	0		ns
T_3	Set-up time from 1.0V core supply before clocks are applied	0		ns
T_4	Reset active time after clocks are applied	1000		ns

Note 1: Active reset is a high power state.

Powering Down.

The core should be powered down first followed by the I/O supplies, or together.

4.2 Synchronization – (Synchronization Master/Slave mode)

The control and synchronization interface for the PC202 device enables basic control of device operation. This interface has three functions; to enable PC202 picoArray device initialization, to support code debug and to enable mutual device synchronization in multiple PC202 systems.

At Reset, synchronization of the picoArray is required to maintain picoBus synchronization between several picoArray devices. This ensures that they start, stop, step and halt together. In a multi-picoArray system a single device is chosen as a Master, the rest being “Slaves”. The Master generates the synchronization clock, run, step and syncro-run signals. Any device however may drive the “halt” signal low and halt all picoArrays in the system. Figure 9 shows the connections required for a dual PC202 Processing system and

Figure 10 gives examples of sync_IF for a stand alone system.

NOTE: For Multiple Processor systems it is recommended to select the Synchronization Master and Boot Master to be the same device.

Figure 9: Sync_IF connections for dual/multi processor system

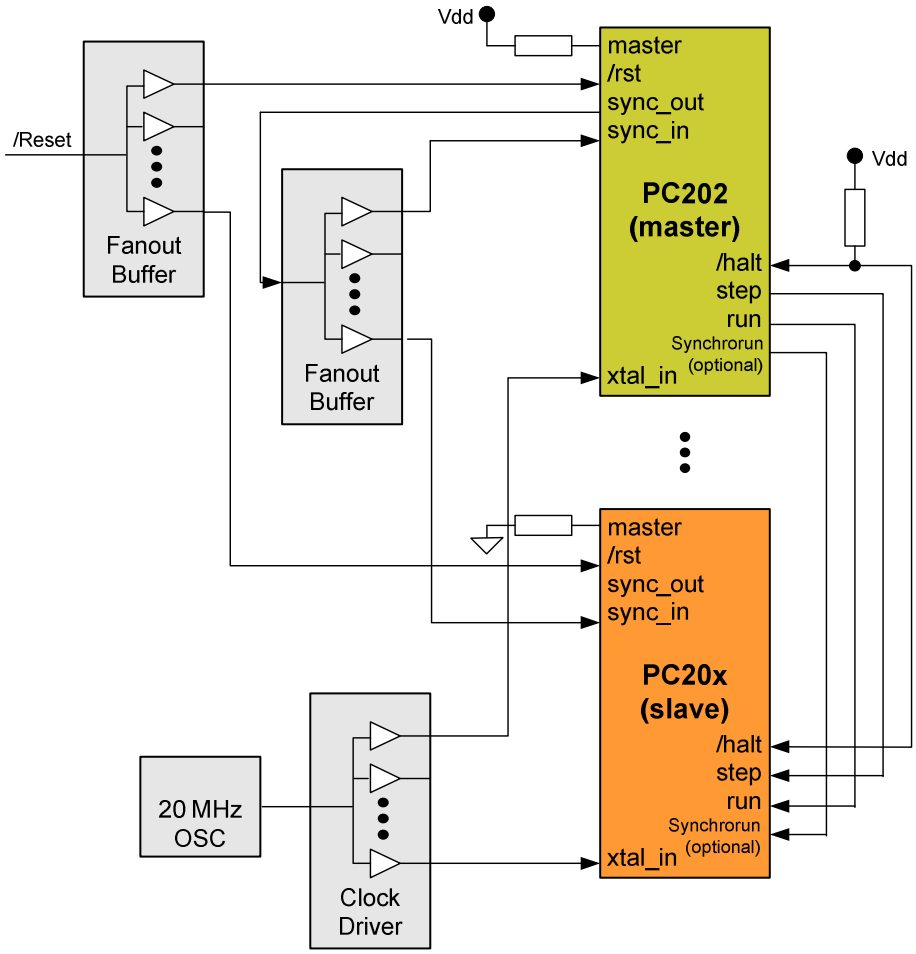
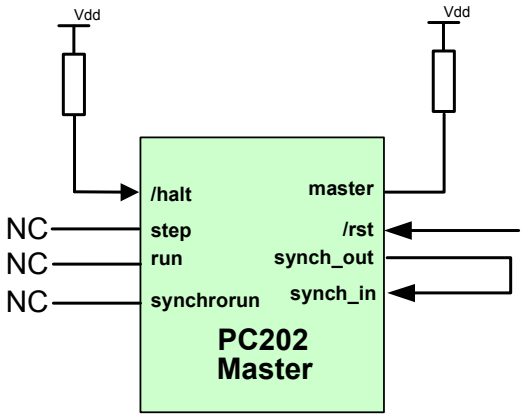


Figure 10: Sync_IF connection for a PC202 system



4.3 EBI (Extended Bus Interface)

The EBI for the PC202 is designed to provide interfaces to external memory and peripherals. These include:

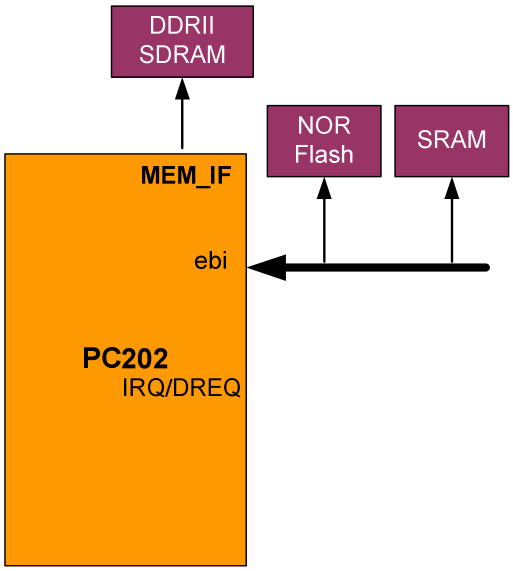
• NOR Flash interfacing

• Asynchronous SRAM interfacing

Please refer to section 5 for the EBI memory Map

The EBI supports interfacing to numerous devices however there is an upper limit to the number of devices which can be coupled to the interface in a glue-less fashion, still allowing the device to boot from NOR Flash devices. The diagram below indicates a possible configuration of the PC202 device in master mode.

Figure 11: Typical EBI configuration.



Capacitive loading of the address, control and data bus by the addition external slave components will impact the pin timings of the interface. If the total capacitive load, on the EBI, **exceeds 50pF** then the device will fail to boot from NOR Flash.

The external bus interface (EBI) has 8 chip select outputs, which decode the memory map. Each of the chip selects at boot up are shown in the following Table 11 The Decode Regions are also Configurable please refer to Appendix O of the PC20x Programming Guide.

Table 11: ebi_decode Memory types

/ebi_decode [7:0]	Default Memory Type	Default Data bus Width
0	NOR Flash	32bit (Note1)(Note 2)(Note 3)
1	NOR Flash	32bit (Note1)
2	Async SRAM	32bit
3	Async SRAM	32bit
4	Async SRAM	32bit
5	Async SRAM	32bit
6	Async SRAM	32bit
7	Async SRAM	32bit

Note1: The default, data bus width can be overridden at boot time by the ebi_dwidth inputs to the PC202. Table 12: ebi_dwidth

Note2: Decode regions 0 & 1 default to NOR flash timings and interface style after power-on/reset.

Note3: Decode regions 0, for the PC202, master mode, must have some NOR flash connected, or the device will not boot.

Table 12: ebi_dwidth

ebi_data width[2:0]	Data bus Width
000	16 bits
001	32 bits
100	8 bits

NOR Flash Interface

The external bus interface (EBI) provides direct interfacing to NOR Flash, support for the following configurations of Flash memory are provided: see Error! Reference source not found. And **Figure 12** for the connection details

Table 13: Suggested NOR Flash Variants.

Number of Devices	Memory(MB)	Bus width
1	64MB	32bit data bus
2	16MB	16bit data bus
4	32MB	32bit data bus
1	32MB	8bit data bus
2	64MB	16bit data bus
1	128MB	32bit data bus
2	32MB	16bit data bus
4	64MB	32bit data bus
1	16MB	16bit data bus
2	64MB	32bit data bus
1	64MB	8bit data bus
2	128MB	16bit data bus
1	64MB	16bit data bus
2	128MB	32bit data bus

Figure 12: Typical NOR Flash Connection

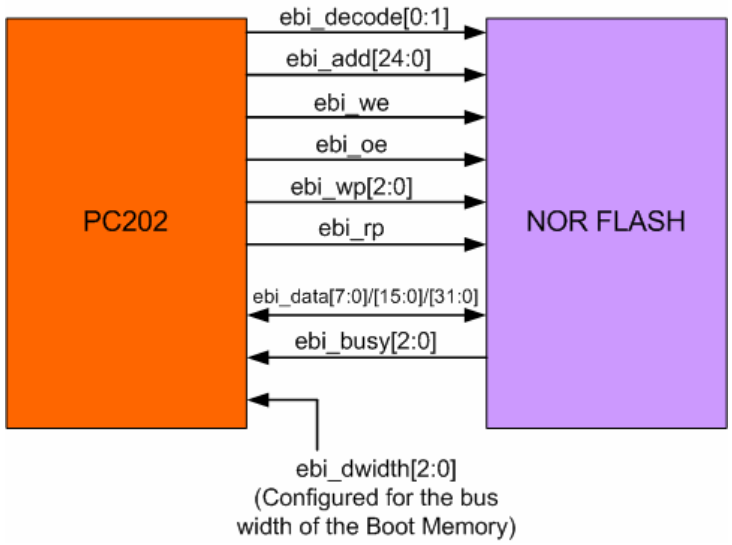


Figure 13: NOR Flash Read Access Timing Diagram

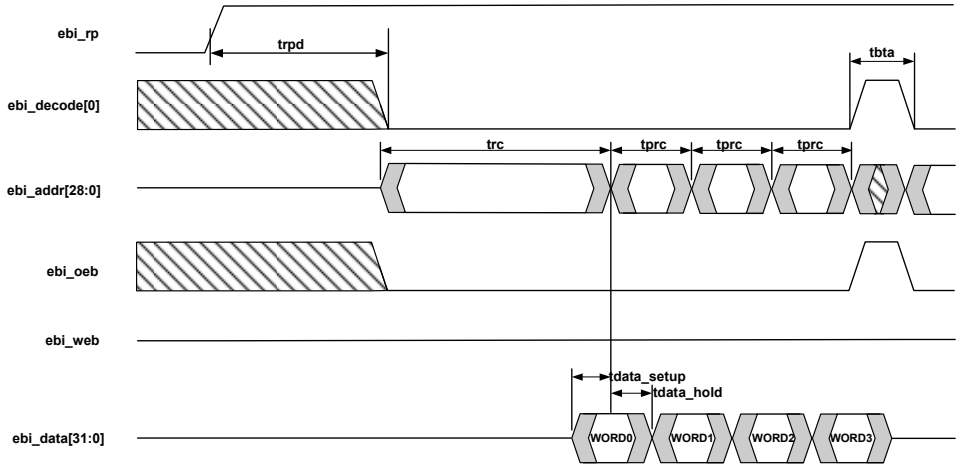


Table 14: Flash timing parameters

Parameter	Description	Min	Max	Units
Trpd	Time from reset/power down exit to start of access	198		ns
Trc	Read cycle time	200		ns
Tprc	Page mode read cycle time	64		ns
Tbta	Bus turn around time at end of read cycle	21		ns
Tdata_setup	Read data setup time	10		ns
Tdata_Hold	Read data Hold time	0		ns

The timings associated with this interface are programmable.

4.3.1 Interfacing to Asynchronous SRAM

The EBI can support asynchronous SRAM devices directly connected. Ebi_decode [7:2] have default values to allow the EBI to access 32 bit wide SRAM without re-programming, provided they have a read access time of 20ns or less.

Figure 14: Asynchronous SRAM Read Access

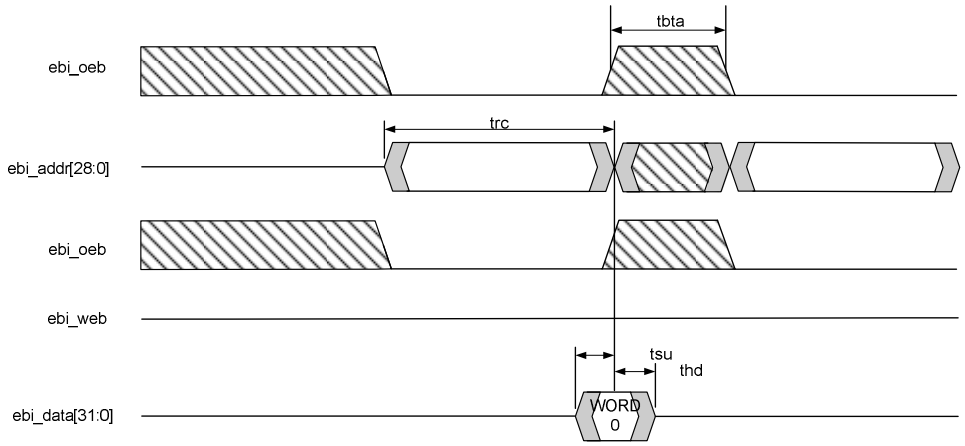


Table 15: Asynchronous SRAM Read Access Timing

Parameter	Description	Min	Max	Units
trc	Read cycle time, address/select to data valid	43		ns
tbta	Bus turn around time for read / write or write / read access.	21		ns
tsu	Read Address Setup time.	8		ns
thd	Read Address Hold time.	5		ns

The timings associated with this interface are programmable.

Figure 15: Asynchronous SRAM Write Access

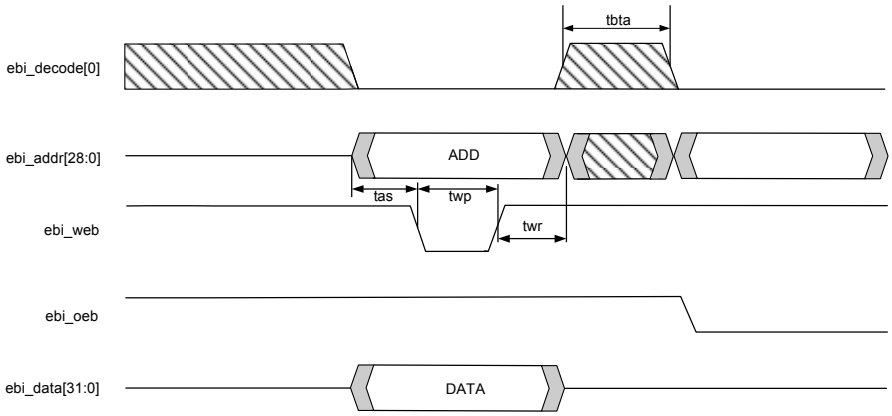


Table 16: Asynchronous SRAM Write Access Timing

Parameter	Description	Min	Max	Units
tas	Address set up time to write pulse driven low	14		ns
twp	Write pulse length	14		ns
twr	Write address hold time	7		ns
tbta	Bus turn around time for write/read access or write/write read access	21		ns
tsu	Read Address Setup time.	8		ns
thd	Read Address Hold time.	5		ns

The timings associated with this interface are programmable.

4.3.2 Interfacing to Synchronous Devices

The EBI can support synchronous accesses. Ebi_decode [7:2] regions can be reconfigured as shown within the PC20x Arm subsystem programming guide. The EBI clock can be configured to run synchronously with the control and data signals. Below are the timing diagrams and parameters for the EBI for synchronous accesses.

Figure 16: Synchronous Read Access

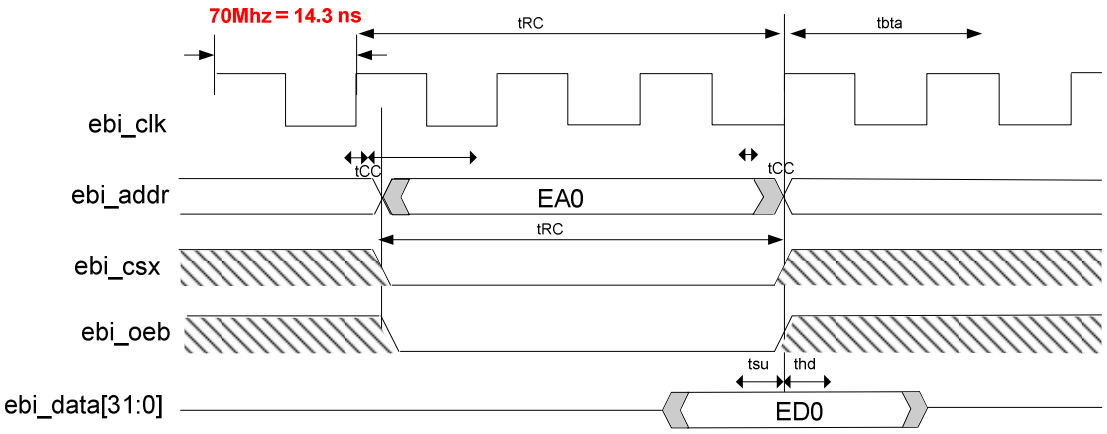


Table 17: Synchronous Read Access Timing

Parameter	Description	Min	Max	Units
trc	Read cycle time, address/select to data valid	43		ns
tcc	Time for control signal generation from ebi rising edge	1	3	ns
tbta	Bus turn around time for read / write or write / read access.	21		ns
tsu	Read Address Setup time.	8		ns
thd	Read Address Hold time.	5		ns

The timings associated with this interface are programmable.

Figure 17: Synchronous Write Access

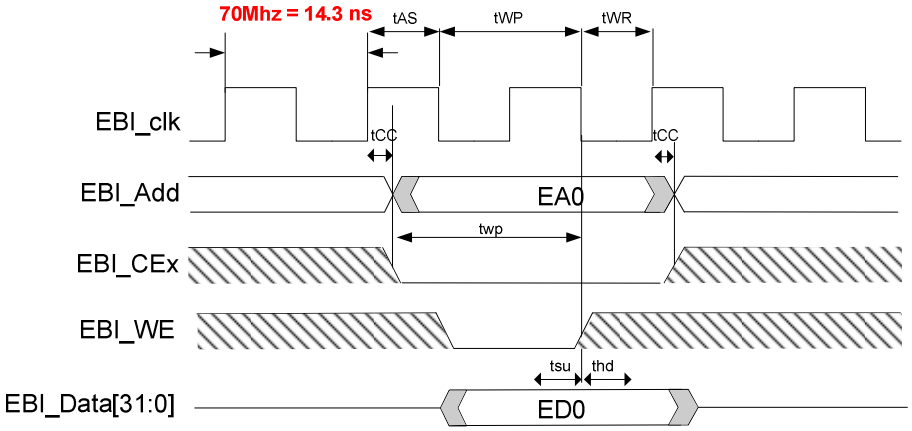


Table 18: Synchronous Write Access Timing

Parameter	Description	Min	Max	Units
tas	Address set up time to write pulse driven low	7		ns
twp	Write pulse length	20		ns
tcc	Time for control signal generation from ebi rising edge	1	3	ns
twr	Write address hold time	7		ns
tbta	Bus turn around time for write/read access or write/write read access	21		ns
tsu	Write Address Setup time.	8		ns
thd	Write Address Hold time.	5		ns

The timings associated with this interface are programmable.

4.3.3 External Bus Address Interface requirements for 8,16 and 32 bit devices

When Connecting the External Bus interface to 8, 16 or 32 bit devices the addressing of these devices needs to be considered. Below details the address connections required for 8,16 and 32 bit devices.

When a decode region is programmed to access an 8bit device:

ebi_addr[0] is equivalent to A(2^0),
ebi_addr[1] is equivalent to A(2^1), etc...

When a decode region is programmed to access a 16bit device:

ebi_addr[0] is equivalent to A(2^1),
ebi_addr[1] is equivalent to A(2^2), etc...

When a decode region is programmed to access a 32bit device:

ebi_addr[0] is equivalent to A(2^2),
ebi_addr[1] is equivalent to A(2^3), etc...

**ebi_addr[0] Automatically aliases to the word boundary.

When interfacing to an 8bit device, the PC202, ebi_addr[0] should be connected to flash_addr[0].

For 16bit & 32 bit devices the PC202, ebi_addr[0] should also be connected to the flash_addr[0]...ebi_addr[n] connected to flash_addr[n].

Further pin descriptions:

ebi_byte_mode:

For access to sub-words and bytes of 16 and 32 bit wide devices, the ebi_byte_mode outputs are used by the External Bus interface. These are Active High. Example below for a 32 bit word:

ebi_byte_mode[0] – first Byte
ebi_byte_mode[1] – 2nd Byte
ebi_byte_mode[2] – 3rd Byte
ebi_byte_mode[3] – 4th Byte

ebi_Ready:

This is used when interfacing to non memory type devices. picoChip do not envisage such interfacing via the ebi. Due to the 3 IPI / ADI interfaces being much more efficient at this type of transaction. If you are wishing to interface the ebi to non memory type devices, please contact support.picochip.com for further details

4.4 Processor interface

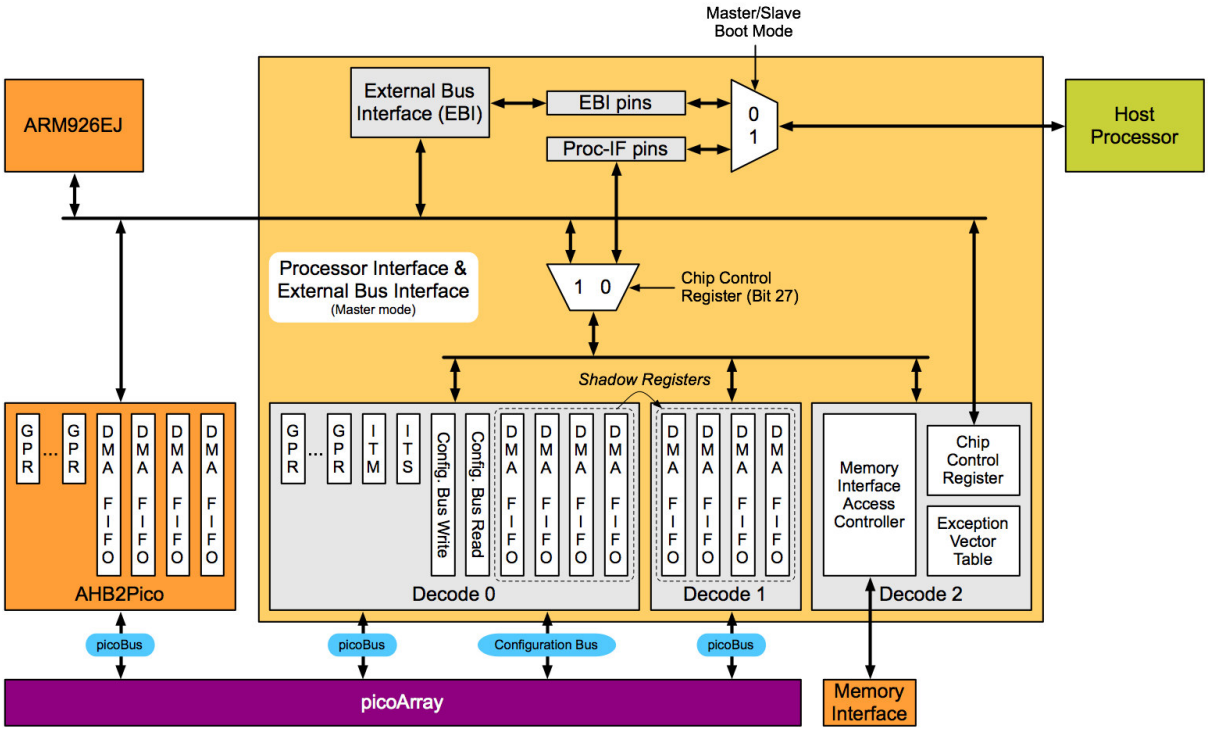
The PC202 microprocessor interface is designed for communications with a external processor. No specific processor family is assumed and data can be exchanged over 8, 16 or 32-bit wide data bus.

The Processor interface supports two basic transaction types

1. **Single (GPR)** – Reading or writing one word at a time.
2. **Burst (DMA)** – Reading or writing words at the same rate as the microprocessor proc clock.

GPR accesses allow access to the majority of memory mapped registers and services within the processor interface, GPR accesses can only be used for single read / write accesses. DMA Accesses are primarily used for the efficient movement of data to and from the picoArray. Accesses to the memory mapped registers and DMA FIFO's are supported within three decode regions. **Error! Reference source not found.** Is a block diagram of the processor interface. It shows the services addressable within each decode region.

Figure 18: Processor interface block diagram.



4.4.1 Processor Interface Services and Decode Regions.

The following Section covers each individual decode region, detailing the memory maps for each region and services available.

Decode Region [0]

Decode region 0 gives access to 32 X 32bit registers the following Services are provided, 24 GPR Registers, Interrupt Mask and status register, Configuration read / write Ports and Four DMA shadow GPR's

General Purpose Register accesses.

The 24 32-bit GPR's can be used by the external microprocessor as a low-rate interface for reading and writing data to the picoArray. Internally, the GPRs either put data onto or get data from the picoBus, or are idle if unused. GPR status registers are also available within the the processor interface which show when new data is available within the GPR. Further details of the GPR status registers can be found within the picoTools user guide.

GPRs can be connected to block or non-blocking signals. When connected to blocking signals data loss is prevented by not allowing new data to be written to the register until the previous data has been read. When connected to non-blocking signals the last value written into the GPR is read back when the GPR is read from.

To avoid losing data when writing to a GPR, the picoBus slot-rate for reading the GPR should be higher than the rate of the data being written by the microprocessor. The slot rate is a power-of-two division of the internal system clock. For example, consider a processor bus running at 75MHz with write access to the same GPR every 20 cycles:

Write update rate = 75MHz / 20 = 3.75MHz.

Maximum Read/write Rate = proc_clk / 8

For example for a proc_clk of 75Mhz the user can write to a gprSrc @ 75 / 8 or 9.375Mhz.

The user should then consider the gprSink they need to connect the signal to on the picoBus the user could attach the signal to the following @ rate – the rate at which the picobus services the proc_if signal.

Minimum slot-rate is @32 = picoArray clock rate / 32 < 9.375 therefore could not be used to service this signal. An @ Rate of 16 could be used to service the signal. 160 / 16 = 10Mhz

NOTE: When writing to a blocking GPR, an overflow error flag will be set if the microprocessor attempts to overwrite data that has not been read. The unread data in the GPR is preserved but the writes are not blocked.

A similar approach should be taken when the microprocessor is reading a non-blocking GPR. To avoid losing data, the picoBus slot-rate for writing to the register should be slower than the rate at which the microprocessor polls it. In this case, the processor may read the same value more than once. For a blocking GPR, the picoBus slot-rate can be faster but the port will be stalled until the processor reads the stored value.

NOTE: When reading from a blocking GPR, an underflow error flag may be set if the microprocessor tries to read the same value more than once.

NOTE: GPR overflow and underflow error flags are maskable.

GPRs are only intended for slowly varying control parameters, which means there is a limit to how fast they can be accessed without losing data. The picoBus @rate should be greater than four times the ratio of the internal system clock to the processor clock. For example, with a 160MHz system clock and a 66MHz processor clock, the fastest rate would be equivalent to an "@10" slot-rate.

Interrupt registers.

The PC202 has two 32-bit interrupt ports for access to an Interrupt Mask (ITM) register and an Interrupt Status (ITS) register. These are accessed in the same way as the GPR ports for read or write access by the microprocessor.

NOTE: The ITS Register is a read only register; bits can only be cleared by the designated Interrupt Controller Array Element.

NOTE: The most significant bit of the ITS register is reserved for the halt interrupt used during debug. The rest are for general-purpose use.

The Interrupt interface includes an IRQ output pin up_irq. This is set according to the following operation:

up_irq = BITWISE NOR { BITWISE AND [ITM register, ITS register] }

There are two IRQ modes of operation: edge or level triggered. Level triggered is the default mode where the IRQ pin stays low until all unmasked ITS bits are cleared. In the edge mode, the IRQ pin will pulse low for each new unmasked ITS bit. The mode and length of IRQ pulse are set during PC202 device configuration.

For a detailed view of the Registers please see the PC20X Programmers Guide Appendix L.

Configuration Bus Read / Write Port

The processor interface provides microprocessor access to the internal PC202 configuration bus. It can be used for device configuration at power up and to read information from the array elements for debug purposes during runtime.

Prior to configuring the PC202, it must be reset to remove any prior configuration data as detailed in paragraph **Error! Reference source not found.**. This places all registers into a known, idle state and disables all array elements. The Configuration Ports use 22 bits of the data bus for write accesses (16bits of data 6 flags) and 20 for configuration reads (16bits of data 4 Flags) (up_data[21:0]) (up_data[19:0]) respectively.

NOTE: Both the up_adhi[1:0] and output enable line /up_oe should be disabled (set high) during device configuration.

To read-back configuration and diagnostic data, the microprocessor must first send a read-request command to the configuration write-port. Data can then be read using single GPR type accesses.

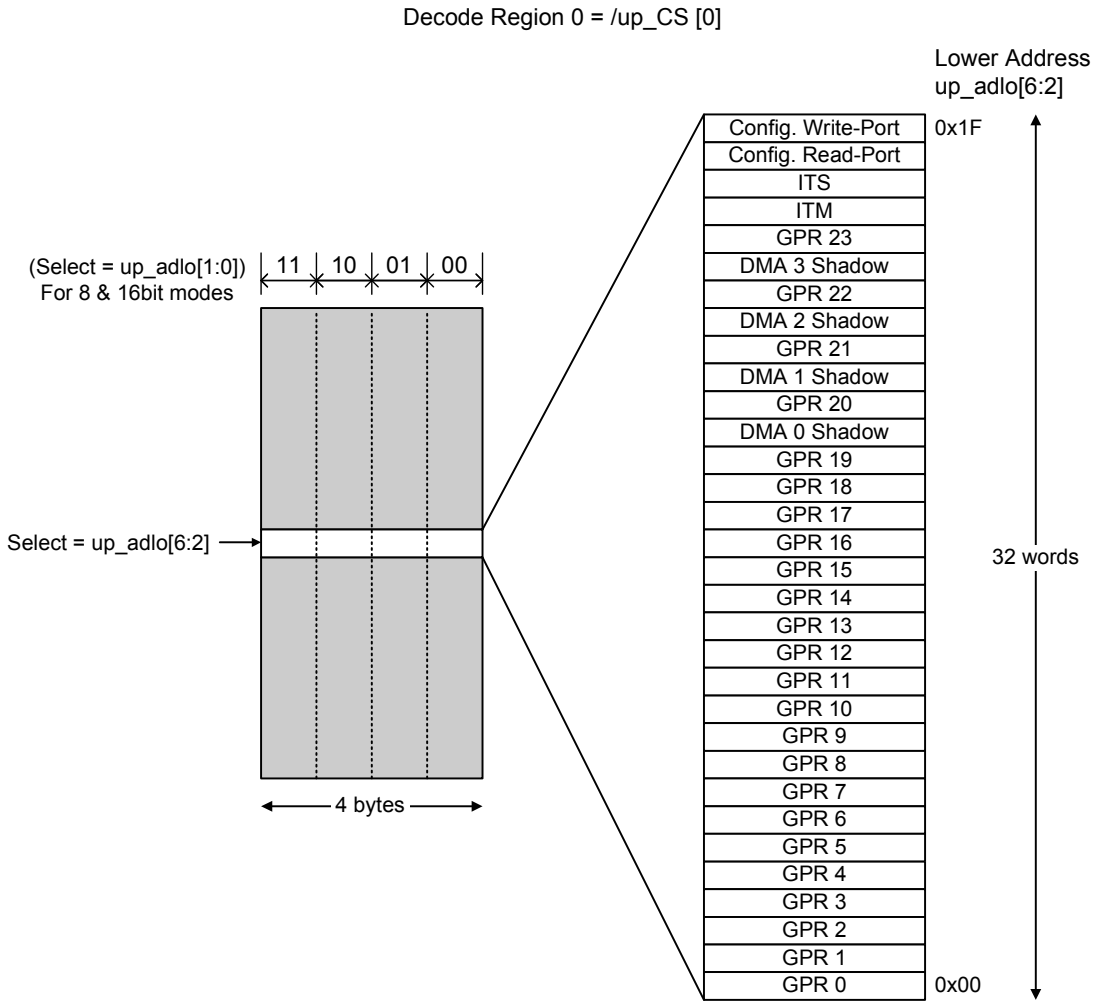
Shadow DMA GPR's

There are four DMA GPR shadow registers available that provide access to the four DMA channels via a simple GPR access. **Error! Reference source not found.**: Show's the address locations for the internal GPR registers. **Error! Reference source not found.** Shows the

Table 19: Address mapping for microprocessor interface Decode region [0]

Lower Address adlo[6:2]		Service	Abbreviation
Dec	Hex		
124	7C	Configuration Bus Read-Port	CBWP
120	78	Configuration Bus Write-Port	CBRP
116	74	Interrupt Mask Register	ITM
112	70	Interrupt Status Register	ITS
108	6C	General-Purpose Register 23	GPR 23
104	68	Shadow GPR for DMA Channel 3	DMA3(GPR)
100	64	General-Purpose Register 22	GPR 22
96	60	Shadow GPR for DMA Channel 2	DMA2(GPR)
92	5C	General-Purpose Register 21	GPR 21
88	58	Shadow GPR for DMA Channel 1	DMA1(GPR)
84	54	General-Purpose Register 20	GPR 20
80	50	Shadow GPR for DMA Channel 0	DMA0(GPR)
76	4C	General-Purpose Register 19	GPR 19
72	48	General-Purpose Register 18	GPR 18
68	44	General-Purpose Register 17	GPR 17
64	40	General-Purpose Register 16	GPR 16
60	3C	General-Purpose Register 15	GPR 15
56	38	General-Purpose Register 14	GPR 14
52	34	General-Purpose Register 13	GPR 13
48	30	General-Purpose Register 12	GPR 12
44	2C	General-Purpose Register 11	GPR 11
40	28	General-Purpose Register 10	GPR 10
36	24	General-Purpose Register 9	GPR 9
32	20	General-Purpose Register 8	GPR 8
28	1C	General-Purpose Register 7	GPR 7
24	18	General-Purpose Register 6	GPR 6
20	14	General-Purpose Register 5	GPR 5
16	10	General-Purpose Register 4	GPR 4
12	0C	General-Purpose Register 3	GPR 3
8	08	General-Purpose Register 2	GPR 2
4	04	General-Purpose Register 1	GPR 1
0	00	General-Purpose Register 0	GPR 0

Figure 19: Address mapping for microprocessor interface Decode region [0].



Decode Region [1]

Decode region 1 gives service access to 4 Memory Mapped 64K byte DMA Channels

Direct Memory Access (DMA)

There are four DMA channels to map a section of the external microprocessor memory to the internal PC202 memory. The memory map in Figure 2021 shows an example with 64k bytes per DMA channel, DMA channels can be mapped to any location within the microprocessor memory space. Each DMA channel has an associated /dreq line section describes the /dreq configuration and behavior in further detail.

Figure 2020 shows the DMA channel configured for write access. A FIFO buffers the data written from the microprocessor to the selected DMA channel. The FIFO output is connected to the picoBus and data is transferred to the internal array elements by using a get command from within the software. A DMA channel configured for read access uses a dual-port RAM for the FIFO to allow for the internal slot-rate writing data to the port being faster than the external processor read clock. This ensures that there is no loss of data.

NOTE: Each DMA channel can be configured for read or write access, but not both.

The number of cycles between contiguous write and read cycles is adjusted via the Acc2Next parameters and the initial read data delay by the Acc2First parameter.

Figure 20: DMA channel configured for write access

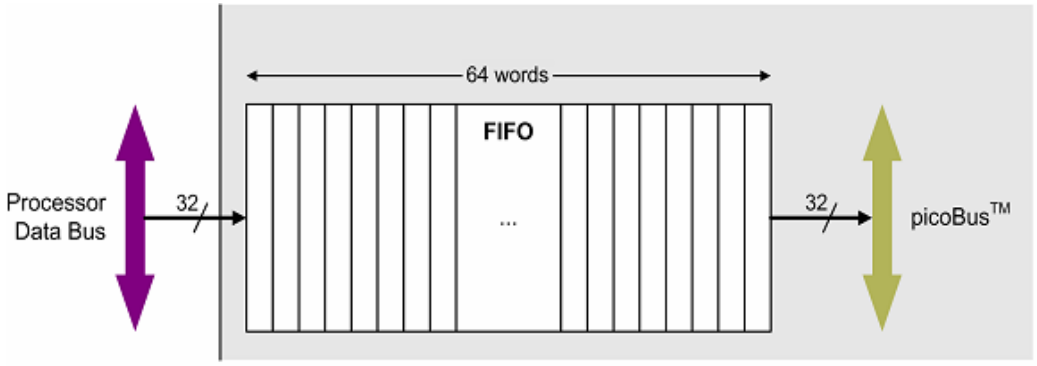
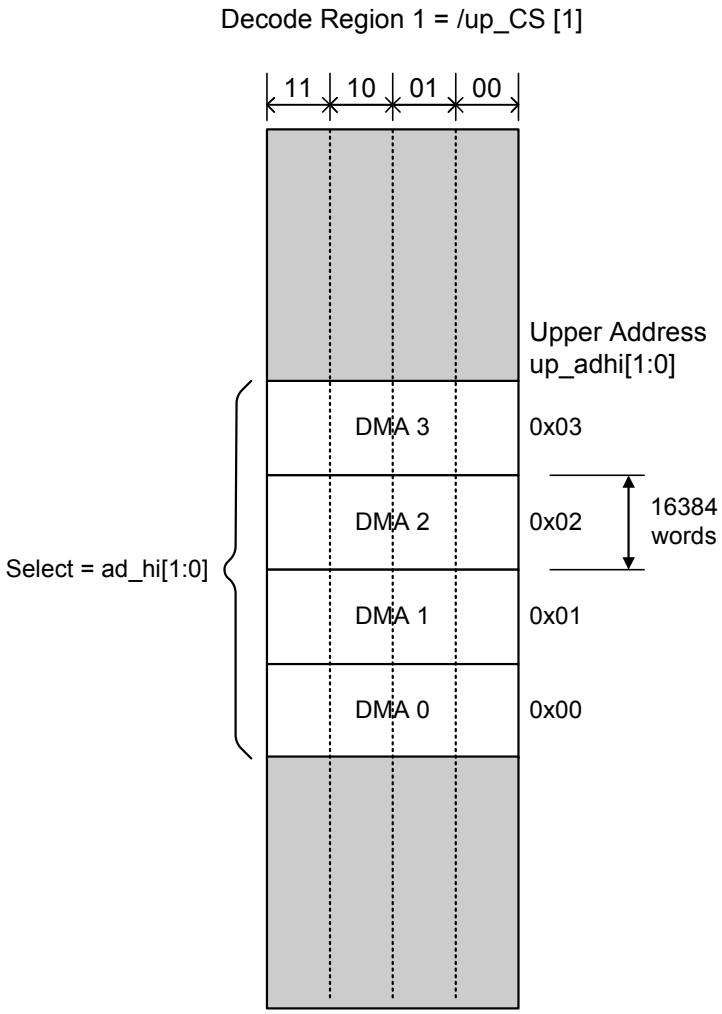


Table 20: Address mapping for microprocessor interface Decode region [1]

Upper Address ad_hi[1:0]		Service	Abbreviation
Dec	Hex		
0	00	Direct Memory Access Channel 0	DMA0
1	01	Direct Memory Access Channel 1	DMA1
2	02	Direct Memory Access Channel 2	DMA2
3	03	Direct Memory Access Channel 3	DMA3

Figure 21: Example of Address lines [17:16] connected to the upper address adhi[1:0] for microprocessor interface Decode region [1] of 64kbyte blocks.



Decode Region [2]

Decode region 2 gives service access to the Chip Control Register and the Memory interface controller.

Chip Control Register

For further information relating to the Chip control register please refer to the PC20X Programmers Guide.

Memory interface Access controller

From the proc_if the external host processor may access the SDRAM attached to the PC202 through the mem_if as well as the internal 128K byte SRAM. This is done through the mem_if Access Controller. The Access Controller is a keyhole port into the memory space controlled by the mem_if arbitration block. It consists of a read FIFO and a write FIFO. Each FIFO has additional control registers. The address register (read and write) sets the address in the mem_if space that the read or write transaction occurs. The FIFO size register determines the FIFO size from 1 to 64 in power of 2 increments (1, 2, 4, 8, 16, 32, and 64). Figure 22 shows a block diagram of the mem_if Access Controller. Details of read / write accesses, configuration registers and method of operation can be found within the PC20X Programmers Guide Appendix B.

Figure 22: Mem_if Access Controller.

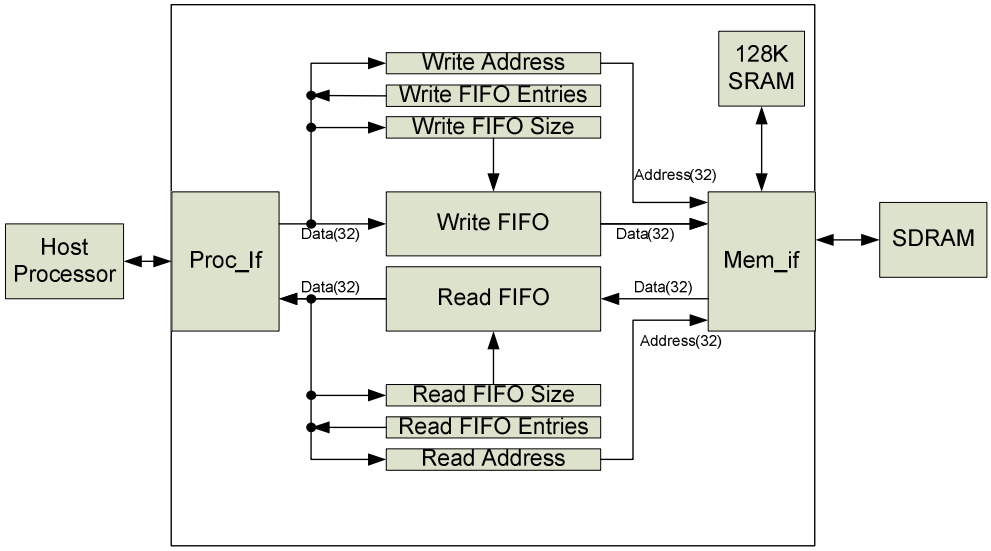
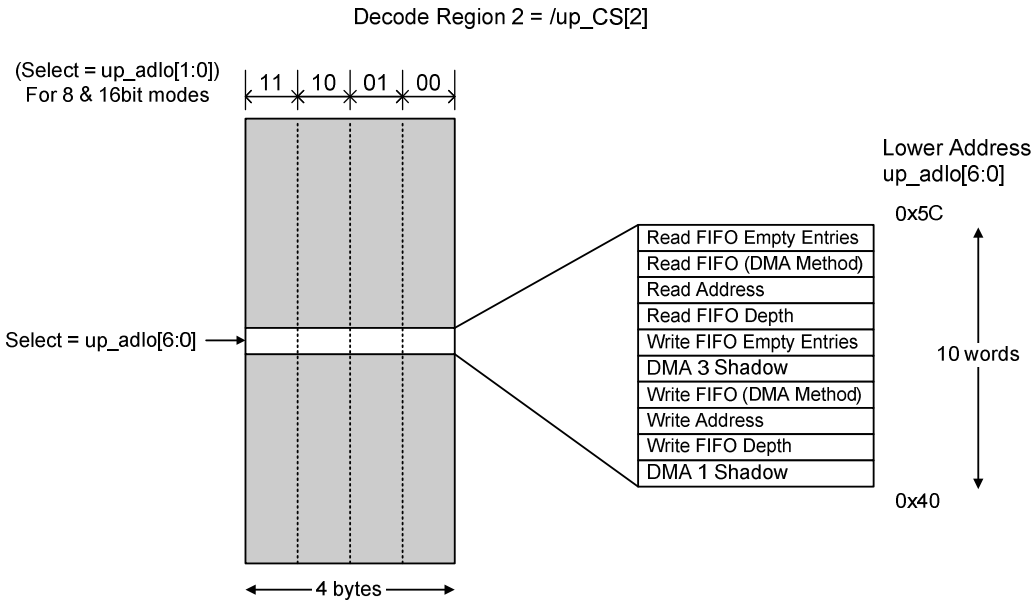


Table 21: Address mapping for microprocessor interface Decode region [2].

Upper Address ad_io[6:0]		Service	Abbreviation
Dec	Hex		
92	5C	Read FIFO Empty Entries	
88	58	Read FIFO (DMA Method)	
84	54	Read Address	
80	50	Read FIFO Depth	
76	4C	Write FIFO Empty Entries	
72	48	Write FIFO (DMA Method)	
68	44	Write Address	
64	40	Write FIFO Depth	

Figure 23: Address mapping for microprocessor interface Decode region [2].



4.4.2 Processor Interface Signals

The three decode regions and ports contained within each region are selected by taking a subset of the processor interface address lines and three chip-select lines: up_adlo[6:0], up_adhi[1:0] and /cs[2:0]. The chip-selects are active-low signals and are typically generated from a combination of the higher significance external address bus lines to give two views of the address space.

1. **32x32-bit registers**
   - up_adlo[6:2] active, up_adhi[1:0] ignored.
   - These are mapped to up_a_reg[6:2] as shown in Decode Region [0].
   - up_adlo[1:0] are used for 16 and 8bit accesses.
2. **Four DMA channels**
   - up_adhi[1:0] active up_adlo[6:2] ignored.
   - up_adlo[1:0] are used for 16 and 8bit accesses.
   - Each DMA channel is mapped to a block of internal memory storage as defined by up_adhi[1:0] in Decode Region [1] FigureXX. For example, to allocate 64k bytes per channel the external byte address lines [17:16] are connected to the upper address lines, up_adhi[1:0]. Byte address lines [15:0] are ignored since the data goes into a FIFO (a combination of the DMA FIFOs and internal memory).

Error! Reference source not found. Details the address lines and the types of accesses used within each region and service.

NOTE: The register select line, up_adlo[6:2], and DMA select lines, ad_hi[1:0], should never be active at the same time.

NOTE: The shadow addresses for the DMA ports allow them to be accessed as if they were a GPR. Please contact support@picochip.com for further details.

Table 22: Microprocessor interface access types.

Signal	Service	up_adlo[6:2]	up_adhi[1:0]	Supported Data bus width & Access mode adlo[1:0]
/up_cs[0] Decode Region [0]	General Purpose Registers (GPR)	up_adlo[6:2]	Ignored	8,16 or 32
	Configuration bus Read Access.	Up_adlo[6:2]	Ignored	8,16 or 32 Only data bits [20:0] are used Single read/write only
	Configuration bus Write Access	up_adlo[6:2]	Ignored	8,16 or 32 Only data bits [22:0] are used Single read/write only
	Interrupt Status (ITS) & Interrupt Mask Registers (ITM)	up_adlo[6:2]	Ignored	8,16 or 32 Single read only
	Access to DMA[3:0] shadow registers (Non DMA Access)	up_adlo[6:2]	Ignored	8,16 or 32 Single are used Single read/write only
/up_cs[1] Decode Region [1]	Access to DMA[3:0] (DMA Access Ports) Each DMA Channel has an associated /dreq[3:0] output signal.	Ignored	11 – DMA[3] 10 – DMA[2] 01 – DMA[1] 00 – DMA[0]	8,16 or 32 Burst read/write mode.
/up_cs[2] Decode Region [2]	Access to the memory interface – SDRAM and internal SRAM	up_adlo[6:2]	Ignored	8,16 or 32 Single read only

DREQ Signals – dreq[3:0]

Each DMA channel has an associated DMA request line output, /up_dreq[3:0], and can be configured to operate in four modes: fixed or variable length transfers with the DMA request (DREQ) line either a pulsed (edge triggered) or level output. DREQ can be configured as active low or high, and is used in conjunction with a FIFO high watermark value N and low watermark value O as summarized in **Error! Reference source not found.** Care should be taken when setting these watermarks to match the ability of the system to prevent FIFO under or overflows. DMA transfers can be contiguous or as multiple accesses.

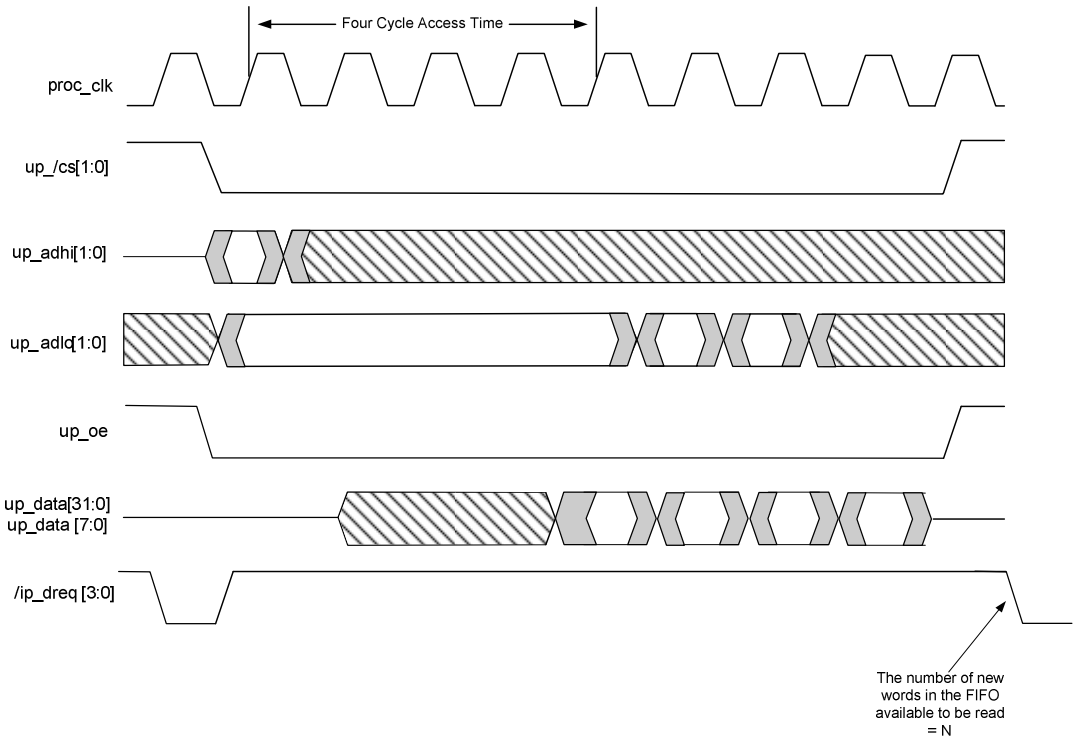
There is a programmable hold-off time for each DMA channel with all of these modes. This is the time, in processor clock cycles, before the DREQ line can be asserted again after it has been de-asserted. There is an associated 10-bit counter for this function.

Table 23: DREQ modes

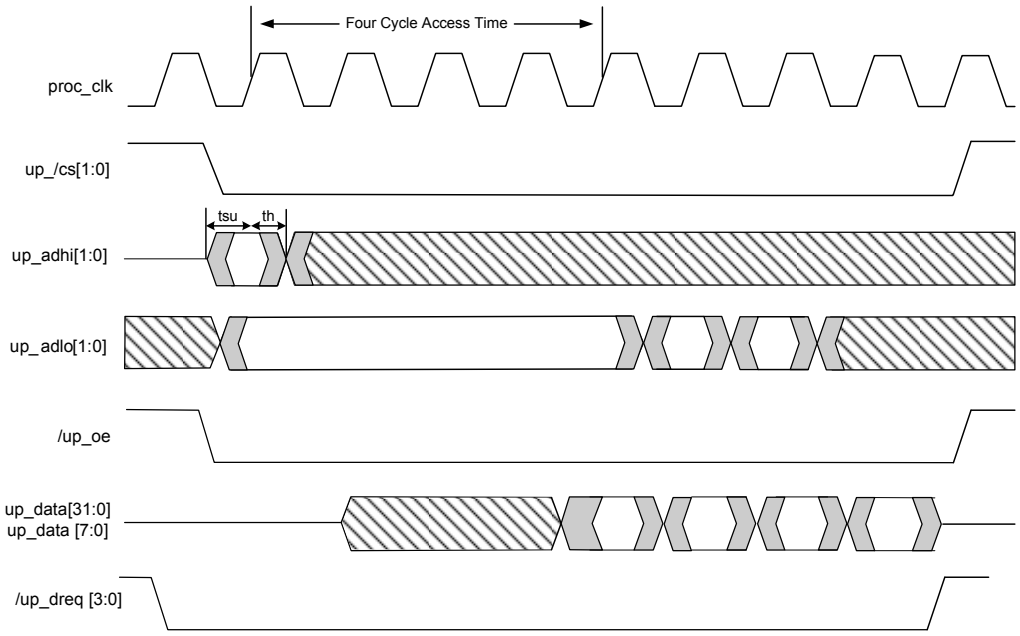
DREQ	Write Access	Read Access
	• DREQ asserted when FIFO contents <= Low threshold Register. “O”	• DREQ asserted when FIFO contents >= High threshold Register “N”.
	Pulsed	
	• DREQ de-asserted when accesses start • DREQ not re-asserted while accesses ongoing	
Fixed	• Number of words transferred must be equal to the value of N the high threshold register before DREQ can be re-asserted.	• Number of words must be equal to the value of O the low threshold before DREQ can be re-asserted
Variable	Any Number of word accesses may be transferred.	
	Level	
Fixed	• Number of words transferred must be equal to N the high threshold register before DREQ is de-asserted.	• Number of words transferred must be equal to O the low threshold register before DREQ is de-asserted.
Variable	• DREQ held active until FIFO contents >= high threshold register	• DREQ held active until FIFO contents <= low threshold register

DMA Examples showing DREQ settings

DMA Read-port access timing (32-bit) Pulsed Output Fixed transfer, length N = 4



DMA Read-port access timing (32-bit) Level Output fixed transfer length N = 4



DMA Read and Write Accesses

DMA accesses can present multiple words per bus transaction at a programmable delay and rate. There are specific registers detailed within the picoTools User Guide “Acc2First” and “ReadAcc2Next, WriteAcc2Next” these configuration registers control the initial data presentation delay, variable from between 4 and 7 “pad_proc_clk” cycles, and the update rate, variable between 1 and 4 “pad_proc_clk” cycles. This flexibility allows slower and/or higher latency devices to use the DMA services of the Processor Interface.
By default the “ Read Acc2First” and “Read Acc2Next” configuration registers present the data after four “pad_proc_clk” cycles and present the next data after a single “pad_proc_clk” cycle. The default value of “WriteAcc2Next” is every “pad_proc_clk” cycle. Figure 15 and Figure 16 show examples of the effects of ACC2First, ACC2Next settings.

Figure 24: DMA read example with ACC2First and ACC2Next settings.

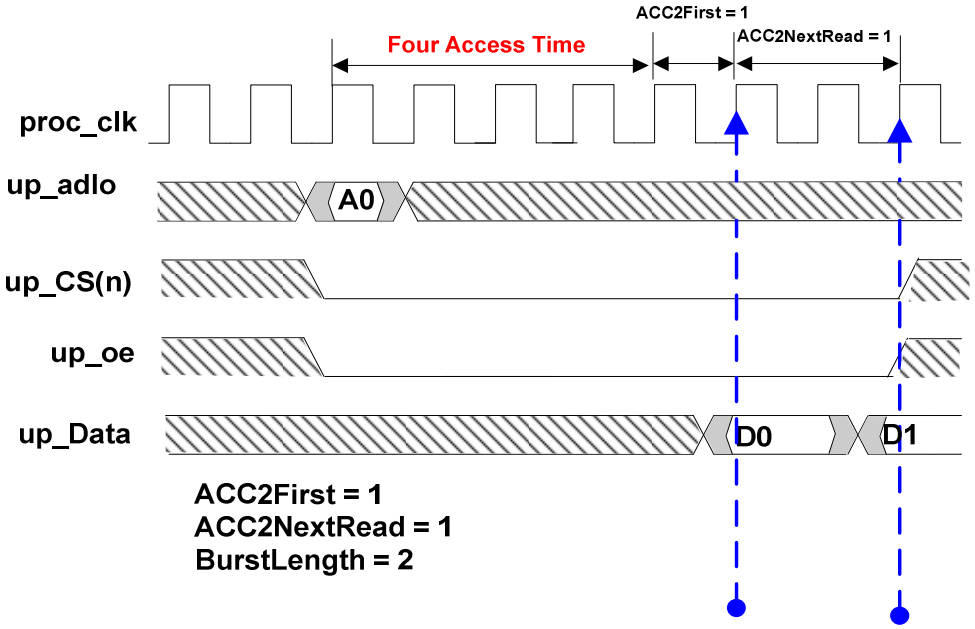
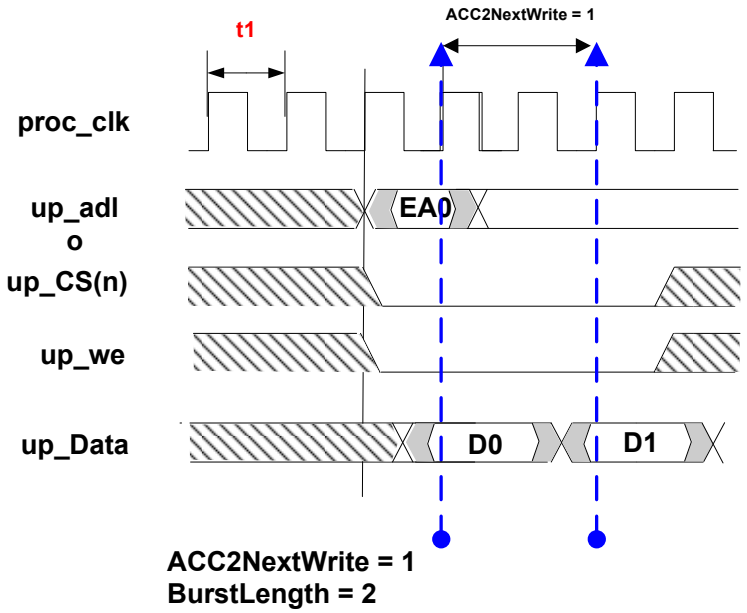


Figure25: DMA Write example with ACC2First and ACC2Next settings.



IRQ Signal

An IRQ signal is available on the processor interface. It is an open-drain signal that requires an external pull-up resistor. The Processor Interface asserts the external signal low. The IRQ can be used with both edge and level sensitive interrupt systems.

Address Bus – up_adlo[6:0] up_adhi[1:0] – 32 bit Accesses.

The address bus is sampled on the first +ve rising edge of "proc_clk". After a chip select "up_CS(n) has been asserted.

When "up_CS(1)" has been asserted "up_adhi[1:0]" are sampled (for 32-bit accesses),

When "up_CS(0)" or "up_CS(2)" are asserted only up_adlo[6:2] are sampled.

Address Bus – up_adlo[6:0] up_adhi[1:0] – 16 / 8 bit Accesses.

When in 16-bit or 8-bit Access mode, the lower up_adlo[1:0] are used, to indicate which byte of half word is currently being presented on the databus. These signals have a more complex relationship with up_CS(n) and the data bus.
For write accesses is occurring up_adlo[1:0] are sampled whenever input data is present. This can be at the end of the transaction for non-burst mode and every acc2next'th cycle in burst mode.
When reading however they also combine to control which 16-bits or 8-bits are placed on the lower bits of the data bus. Therefore can be used by the Host processor to indicate the byte or halfword currently presented on the bus.

Byte Mode Signals – up_mode[2 :0]

The mode pins, up_mode[2:0], set the data-word length for a read or write access. Since all data transfers between the microprocessor and PC202 are assumed to be 32-bits, multiple transfers are required to fill the 32-bits: two in 16-bit mode and four in 8-bit mode. The mode settings are summarized in Table 24

Table 24: Settings for 8, 16 and 32-bit data transfer modes

Data Word Size up_mode[1:0]	Data Word Order up_mode[2]	Data Word Address up_adlo[1:0]	Data Bus
1x = 32-bit	x	Xx	up_data[31:0]
01 = 16-bit	0 = Upper 16-bits last 1 = Lower 16-bits last	0x = Lower 16 bits 1x = Upper 16 bits	up_data[15:0]
00 = 8-bit	0 = Most significant byte last 1 = Least significant byte last	00 = Least significant byte 01 10 11 = Most significant byte	up_data[7:0]

The different modes operate as follows:

- **32-bit Mode** – The data occupies all 32-bits of the data bus and the two least significant bits of the lower address bits are ignored (up_adlo[1:0]).
- **16-bit Mode** – The data occupies the lower 16-bits of the data bus. The 32-bit transfer is completed when the second 16-bit data word is sent or received.
- **8-bit Mode** – The data occupies the lower 8-bits of the data bus. The 32-bit transfer is completed when the fourth byte is sent or received.

NOTE: For software applications that support 8-bit or 16-bit data words, the transfer does not have not have to be made up to 32-bits. For example, by only sending the final data word (set by the data-word order bits in Table 24, the transfer will be completed every time.

NOTE: If using 8-bit data words, the microprocessor can configure the PC202 more efficiently by only sending three out of the four bytes for each transfer – the most significant byte is always ignored.

Direction Signals - /up_oe and /up_we

The "/up_oe" and "/up_we" signals are used to indicate the transaction direction and respectively the driving of the data bus by the Processor Interface and the validity of data. Only one of these signals can be asserted per bus transaction.

4.4.3 Processor Interface timing diagrams.

4.4.3.1 Processor read Access

Figure 26: read-port Access timing (32-bit)

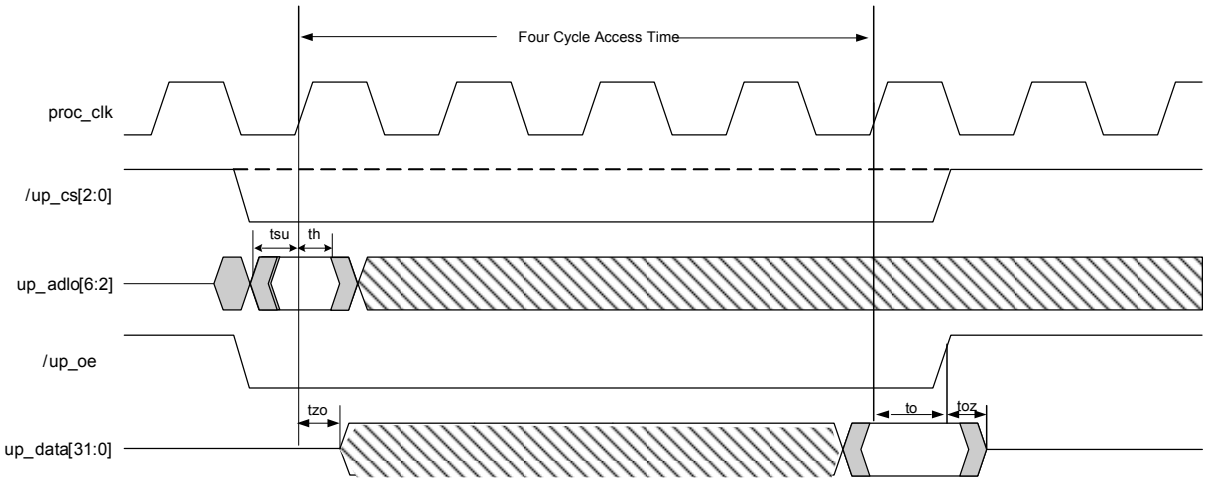


Table 25: Single read timing parameters

Parameter	Description	Min	Max	Units
tperiod	Microprocessor Proc clock period (Note 1)	10		ns
tsu	Set-up time for address and control lines to + edge	2.59		ns
th	Hold time for address and control lines		0.4	ns
tzo	Time from the rising edge of clk to data becomes valid	2.9	8.6	ns
to	Data valid output from the clock rising edge to output-enable rising edge	2.76	6.96	ns
toz	Time before Data becomes High-impedance after rising edge of /up_oe	2.5	5.7	ns
taccess	Time to read first dataword = Four Cycles (Note 2)	40		ns

Note 1: The maximum frequency is 100Mhz.

Note 2: Can be lengthened via the read Acc2First parameter (see picoTools for further ACC2Next & ACC2First Details)

4.4.3.2 Processor 16&8 bit read Access

Figure 27: read-port Access timing (16 & 8-bit)

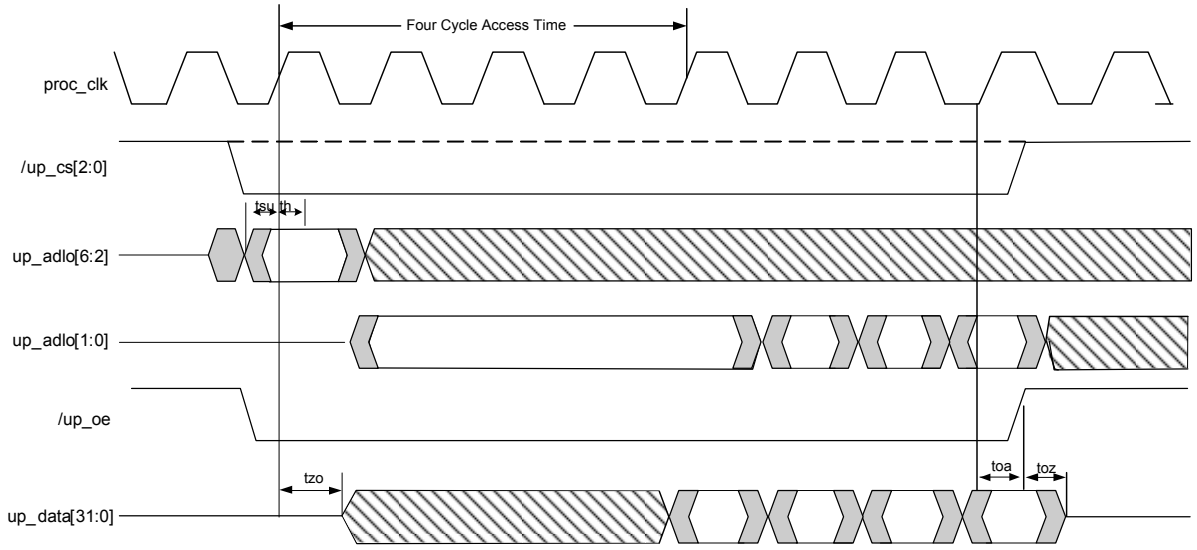


Table 26: Single read timing parameters

Parameter	Description	Min	Max	Units
tperiod	Microprocessor Proc clock period (Note 1)	12.5		ns
tsu	Set-up time for address and control lines to + edge	2.59		ns
th	Hold time for address and control lines		0.34	ns
tzo	Hold time for data output from the output-enable rising clk edge	2.9	8.59	ns
to	Delay for data output from the clock rising edge	2.82	10.27	ns
toz	Hold time for data output from the output-enable edge	2.62	5.74	ns
toa	/up_lo to data time	3.22	8.89	ns
taccess	Time to read first dataword = Four Cycles (Note 2)	50		ns

Note 1: The maximum frequency is 80Mhz.

4.4.3.3 Processor write Access

Write operations are indicated by the assertion of one of the “up_CS[2:0]” signals low in conjunction with the assertion, low, of the “up_we” signal. The “up_addlo[6:2]” signals are sampled on the first clock edge.

Figure 28: Single write-port Access timing

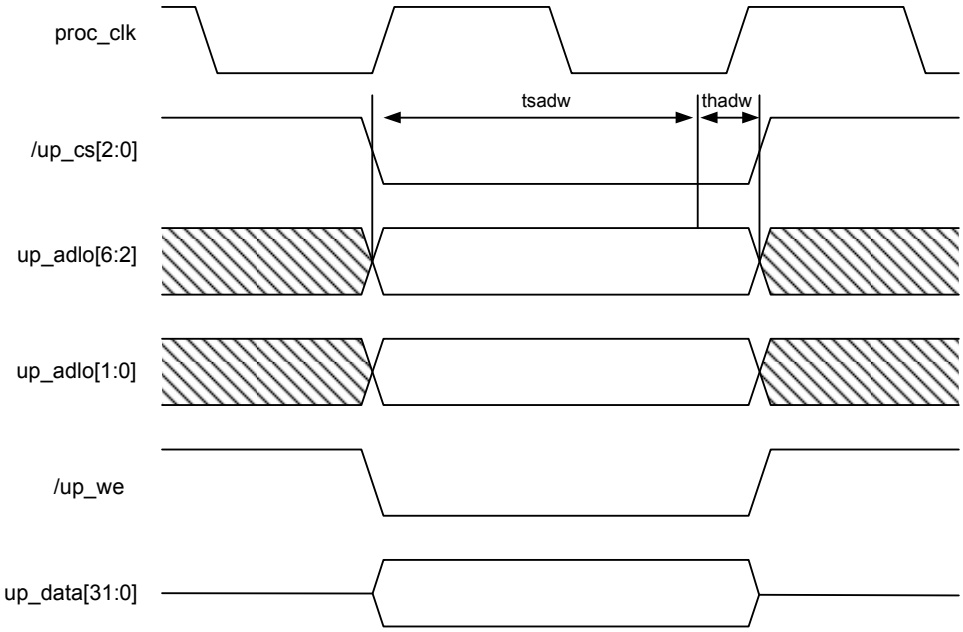


Table 27: Single write timing parameters

Parameter	Description	Min	Max	Units
proc_clk	Microprocessor clock period (Note 1)	10		ns
tsadw	Set-up time for address, data and control lines		2.59	ns
thadw	Hold time for address, data and control lines	0.4		ns

Note 1: The maximum frequency is 100Mhz.

4.4.3.4 Processor DMA Read Access

Figure 29 Example of a DMA transaction with various combinations of CS and up_we. With the programming of Acc2next the user can also delay subsequent data transactions.

Figure 29: read-port DMA Access timing

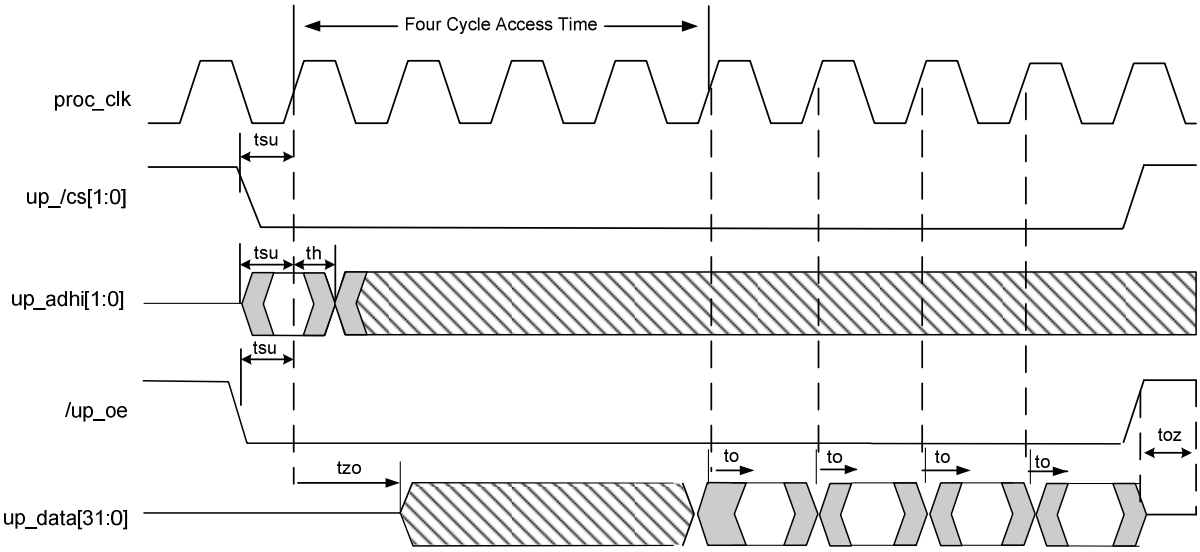


Table 28: DMA read timing parameters

Parameter	Description	Min	Max	Units
tperiod	Microprocessor Proc clock period (Note 1)	10		ns
tsu	Set-up time for address and control lines to + edge	2.59		ns
th	Hold time for address and control lines		0.4	ns
tzo	Time from the rising edge of clk to data becomes valid	2.9	8.6	ns
to	Data valid output from the clock rising edge to output-enable rising edge	2.82	10.27	ns
toz	Time before Data becomes High-impedance after rising edge of /up_oe	2.62	5.7	ns
taccess	Time to read first dataword = Four Cycles (Note 2)	40		ns

Note 1: The maximum frequency is 100Mhz.

Note 2: Can be lengthened via the read Acc2First parameter (see picoTools for further ACC2Next Details)

4.4.3.5 Processor DMA write Access

Figure 30 Shows an example of a DMA transaction with various combinations of CS and up_we. With the programming of Acc2next the user can also delay subsequent data transactions.

Figure 30: write-port DMA Access timing (32bit)

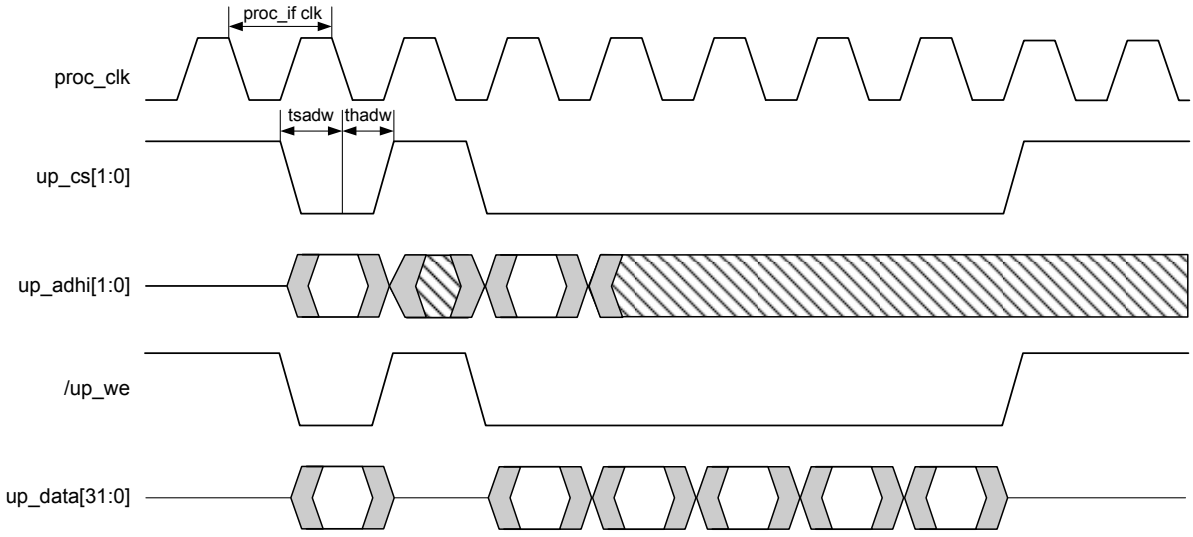


Table 29: DMA write timing parameters

Parameter	Description	Min	Max	Units
proc_if_clk	Microprocessor clock period (Note 1)	10		ns
tsadw	Set-up time for address, data and control lines		2.6	ns
thadw	Hold time for address, data and control lines	0.4		ns

Note 1: The maximum frequency is 100Mhz.

4.5 Interrupt Registers

ITS and ITM Interrupt Registers

There are two interrupt ports and an IRQ pin accessable via the configuration bus and picoBus. The two 32–bit registers are synchronou just like GPRs, but their application is specific – one for Interrupt Mask (ITM) and one for Interrupt Status (ITS). The IRQ pin is the (inverse of the) OR of all the individual Status Bits bitwise ANDed with the Mask bits.

The most significant status bit is reserved for the (debug mode only) Halt Interrupt, the remaining 31 are general purpose.

There are two IRQ modes – Level Triggered and Edge Triggered.

In both modes, the processor can read and write both the Mask register and the Status register. The ITS bits can be reset directly (for faster turnaround, before actually dealing with the interrupt source). Writing a 32 bit word to the ITS address will reset to 0 the individual ITS bits corresponding to any 1s in the 32 bit word. For example, if you wanted to reset to zero ITS[3], you would write 0x0000_0008 to the port.

In Level Triggered mode, the IRQ pin is the “OR” of all the ITS bits, bitwise “AND”ed with the ITM bits. The IRQ pin will stay active until all unmasked interrupts are cleared.

In Edge Triggered mode, the IRQ pin will pulse for each new, unmasked ITS bit. The length of the pulse is configurable.

The IRQ pin is active LOW. Level triggered mode is the default.

The picoArray can write (only) to the Status register but can read (only) the Mask register.

4.6 External Memory Interface

SDRAM DDRII

The SDRAM memory interface provides access to off-chip SDRAM for the picoArray™. For further details regarding the Memory map and configuration registers associated with the SDRAM interface please refer to the PC20X Programmers Guide and picoTools™ User guide.

The SDRAM interface supports various memory configurations. External SDRAM DDR2 devices must operate at a minimum of 125MHz clock frequency, either 256Mbit or 512Mbit sized devices, with 4 banks, and either x8 or x16 bit data bus widths.

(ie a minimum of 2 devices are required to build a 32bit data bus, (but one device for a 16bit data bus). The following configurations are therefore supported in **Error! Reference source not found.**

Table 30: SDRAM Device configuration

PC202 Data Width	Device Density	Device Data Width	Devices Required	Total Density
16 bit	256Mb	x16	1	256 Mb
16 bit	512 Mb	x8	2	1 Gb
16 bit	512 Mb	x16	1	512 Mb
32 bit	256 Mb	x16	2	512 Mb
32 bit	512 Mb	x16	2	1 Gb

Individual SDRAM devices of 1Gbit densities and above, which have 8 banks, are not supported. Devices with x4 data bus width are not supported.

Note: that although individual devices of size 1Gbit are not supported, it is possible to have a total SDRAM density of 1Gbit by using 2 16bit x 512Mb devices to build a 32bit data bus.

The required data bus width (16 or 32bit) and device density (determines row/column bits) are set via a configuration register during the system initialization. See PC20X Programmers guide and the picoTools™ User guide.

PC20x DDRII supported features

The PC20x SDRAM controller supports the following features specific to DDRII SDRAM interfacing.

ODT – ODT (On-Die-Termination). The SDRAM controller has the option of ODT for the following signals. Mem_data[0:31], mem_dqm[0:3], mem_dqs[0:3]. ODT can be disabled or set to one of two drive strengths 75 and 150 Ohms Only.

DQS – The DQS strobes are differential.

Frequency of operation – The minimum clock frequency that the Memory modules can operate at is 125MHz (250MHz data rate). The maximum Frequency is 200MHz (400MHz data rate) In general the slowest memory speed grade parts available operate at 200MHz. The clock frequency is set by the PLL, this internally generated clock value is set to is a multiple of 10 x the Clock source of 20MHz.

Handling of /mem_shield, mem_vref and ODT

Vref is to be equal to 50% of (vddadi) Vref must track changes in vddadi within the recommended range given by your chosen memory module. Typical values are 1 – 2% of vddadi.

It is critical to keep Vref free of noise, as setup and hold margins can be significantly reduced when Vref is noisy. To assist with noise reductions the SDRAM interface has a /mem_shield pin this is used as a shield pin for the noise-sensitive DDR2 threshold reference input mem_vref.

/**mem_shield** should be connected down to the 0V ground plane by a short direct connection. It's important that the mem_vref input (which has a DC bias voltage of VDD_MEM/2) to reduce AC noise, an X7R cap of minimum 10nF should be connected from mem_vref down to 0V and located as close as possible to mem_shield and mem_vref. These two signals have been allocated to balls N22 and P22 on either side of the package centre line, so that the decoupling cap can be put in the gap right next to N22 and P22 on the underside of the PCB footprint.

ODT – If used the On Die Termination pin should be handled like a bidirectional signal when distributing to multiple memory modules.

Figure 31 : PC202 DDRII SDRAM Write timing parameters

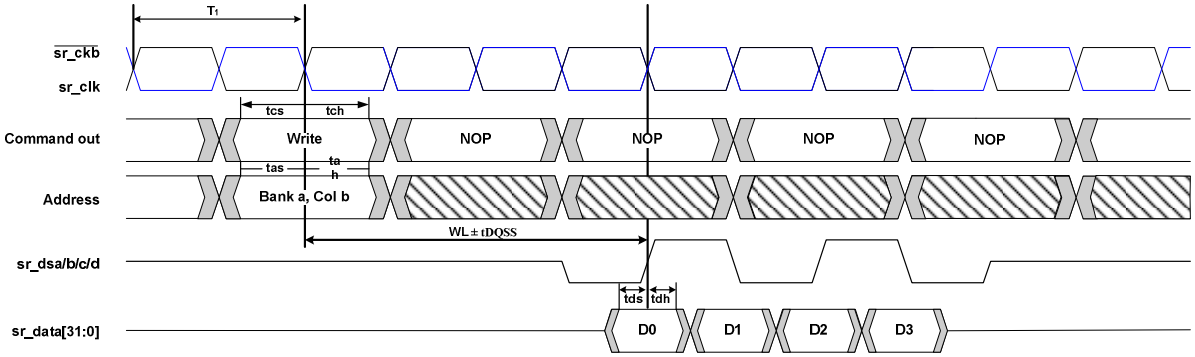


Table 31: PC202 DDRII SDRAM Write timing parameters

Parameter	Description	Min	Max	Units
T1	Write Clock Period	5	8	ns
tas	Address setup time	1.4		ns
tah	Address Hold Time	1.4		ns
tcs	Command setup time	1.4		ns
tch	Command Hold Time	1.4		ns
tds	Data setup time	0.65		ns
tdh	Data Hold Time	0.65		ns
tDQSS	DQS to Clk Skew	-0.7	0.7	ns

Figure 32: DDRII SDRAM Read timing parameters

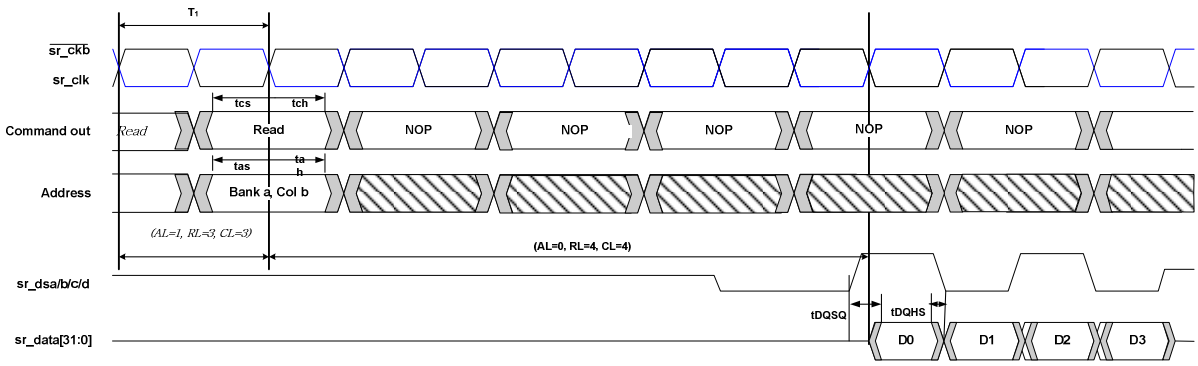


Table 32: DDRII SDRAM Read Latency parameters

Parameter	Description	Min	Max	Units
T1	Write Clock Period	5	8	ns
tas	Address setup time	1.4		ns
tah	Address Hold Time	1.4		ns
tcs	Command setup time	1.4		ns
tch	Command Hold Time	1.4		ns
tDQSQ	Max time valid data following DQS edge		0.7	ns
tDQHQ	Minimum Hold time valid data from DQS Edge		0.7	ns

The Following Diagrams show how consecutive read and consecutive write transactions should appear. For other permutations, for example precharge cycles and read followed by write etc, please refer to your chosen SDRAM Datasheet.

Figure 33: DDRII SDRAM Consecutive Write timing parameters

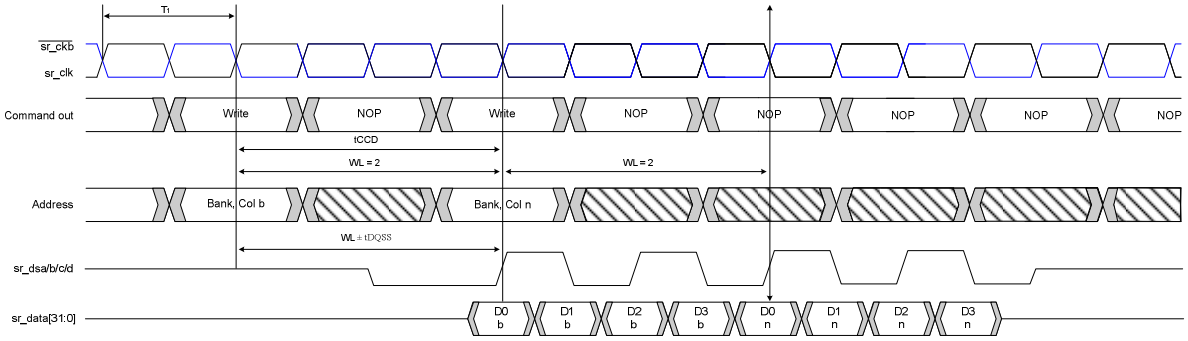
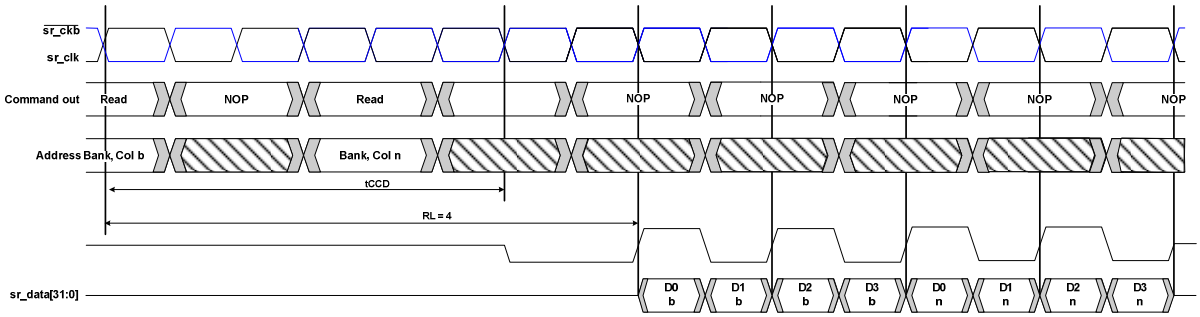


Figure 34: DDRII SDRAM Consecutive Read timing parameters



4.7 Asynchronous Data Interface (ADI)

The ADIs can be used to connect the picoArray™ to a variety of data source or sink devices such as ADCs, DACs or SERDES devices. An external strobe (clock) is used by the ADI to register incoming data, or drive outgoing data. A sync/valid signal is provided in order to support framed or validated data formats.

Each ADI has two 16 bit I/O channels, each with its own strobe and sync/valid signal.

The channels can operate independently as inputs or outputs. Their strobe signals are used to clock the data and sync/valid signals in or out of the ADI. The direction and positive or negative edge clocking can be selected, at configuration time.

synchron User guide provides detailed descriptions of how to use the ADI in various modes including packing and I/Q swapping options.

Data rates

Data can be clocked into or out of the ADI at a maximum of 160MHz. So if only a single edge is used the strobe frequency can be up to 160MHz; for the double data rate case it can be up to 80MHz.

Fsys = 160Mhz

0MHz < strobe frequency < [F_{SYS}] for Single Data Rate.

0MHz < strobe frequency < [$F_{SYS/2]}$ for Double Data Rate.

There is no lower bound on the data rate. The upper bound on the strobe rate approaches the internal system clock. This is a hard limit that applies at all times.

The valid signal can be used to validate input data samples. When validation is not used all data is assumed valid. When the ADI is used as an output the valid signal indicates that the ADI has run out of output data, allowing the ADI strobe to operate above the supplied data sample rate.

When the ADI is used as an input, the AE(s) sinking data must perform GET operations at a rate exceeding the data sample rate in order to prevent data loss. Conversely, when used as an output, the AE(s) sourcing data must perform PUT operations at a rate exceeding the external strobe frequency in order to prevent gaps in the output data. This is especially important when using ADCs or DACs where every sample must represent valid data.

The ADI data path contains FIFOs to accommodate the short term effects of differential clock jitter and burstiness of the internal AE bus communications, with the result that the restrictions above need only be true when considered over any 32 sample periods. The FIFO level can be monitored by performing a configuration bus read (see section on “Monitoring FIFO levels).

Whether used as an input or output, the ADI controls the stalling behavior of its associated Aes and hence the data processing rate within the array.

Sync/Valid Signal

The sync/valid signal is a dual purpose pin which can be configured as either a sync signal or as an (active high or active low) valid signal.

The primary role of the sync/valid signal is to provide some means of associating input or output data with specific picoBus™ data. It also allows some flexibility in I/O data rates by allowing the ADI to disregard non-valid input data or indicate non-valid output data when the FIFO underflows. This is important for interfacing to devices using framed data structures or SERialiser/DESerialiser chips.

The sync/valid signal can be used to validate input data samples. Invalid data is discarded right at the input, which reduces the effective input data sample rate. When validation is not used, the data sample rate equals the strobe frequency. When the ADI is used as an output the valid signal indicates that the ADI has run out of output data, allowing the ADI strobe to operate above the data sample rate.

The function and polarity of the valid signal is configurable. If the sync/valid functions are not required, the signal should be tied high or low via a 4K7 resistor and the ADI configuration registers programmed so that this has the desired effect.

picoBus™ Stream (Hardware Port) Multiplexing

When an ADI channel is used as an input, the data can be de-multiplexed into a number of parallel picoBus™ streams under control of the sync/valid signal, which is an input in this configuration. The ADI supports 1:2, 1:3, 1:4, 1:5, 1:6, 1:7 and 1:8 de-multiplexing modes, permitting up to 8 picoBus™ streams.

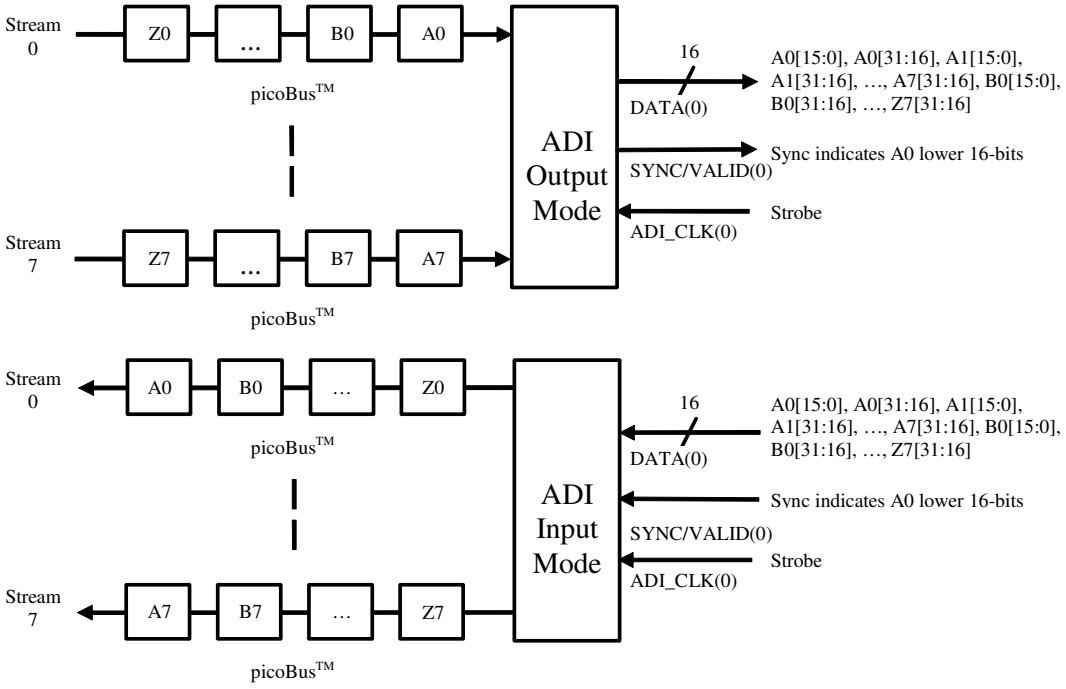
When an ADI channel is used as an output, the data is multiplexed from a number of hardware ports into a single output stream accompanied by the sync/valid signal, which is an output in this configuration. As in the input mode, the ADI supports 1:2, 1:3, 1:4, 1:5, 1:6, 1:7 and 1:8 multiplexing modes.

Note that the multiplexing mode is independent of the data packing mode. This allows a number of different data packing schemes to be used in conjunction with multiplexing.

An example of the multiplexing mode is shown in Figure 35.

Note: The sync/valid pin is being used in sync mode in this example. If the sync signal is active it re-synchronizes data to denote the first data value (i.e. lower 16-bits internally) of stream 0.

Figure 35: picoBus™ port multiplexing mode (with sequential packing)



Data Packing and Unpacking

The various packing modes are described below.

Sequential Packing

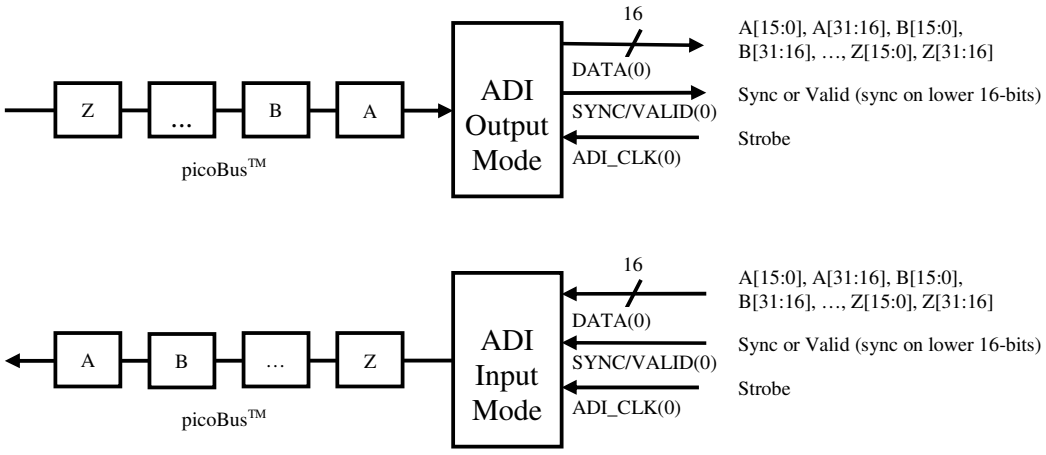
Consecutive 16-bit words sequentially occupy least and most significant halves of the picoBus™ signal(s). This is shown in Figure 36.

Note: that the sync/valid pin may be used in sync or valid mode, however in sync mode the sync signal only affects port multiplexing and has no effect on packing. In valid mode it denotes no data is present.

In sync mode, if the sync signal is active it re-synchronizes data packing to denote the first data value (i.e., lower 16-bits internally).

In valid mode, it denotes no data is present.

Figure 36: Sequential Packing Mode



IQ Packing

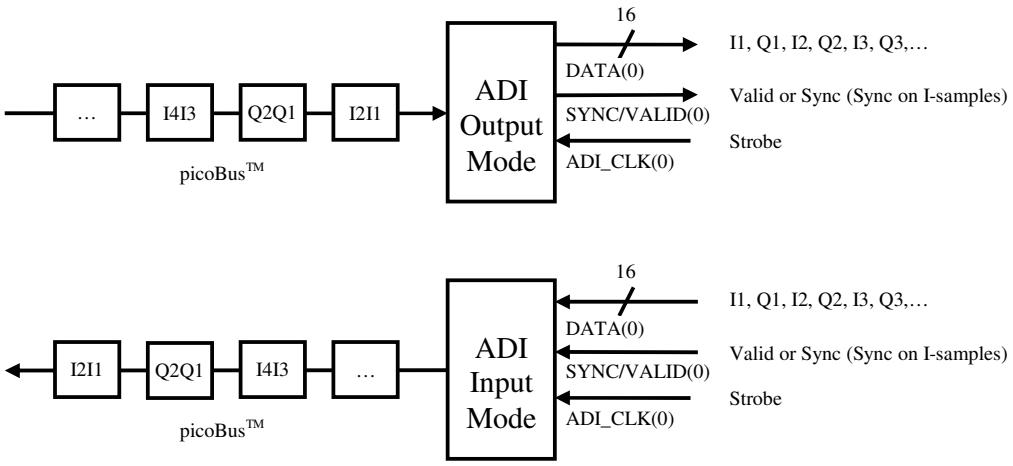
Consecutive 16 bit words are re-ordered so that pairs of odd and even numbered words are presented sequentially. This is shown in Figure 37.

Note that the sync/valid pin may be used in sync or valid mode.

In sync mode, if the sync signal is active it re-synchronizes data packing to denote the I sample (i.e., lower 16-bits internally).

In valid mode, it denotes no data is present.

Figure 37 IQ Packing Mode



The effect of sync/valid on packing.

Packing Option	Direction	Function of Sync	Function of Valid
Sequential packing	I/P	Packing resynchronised to place sync data in bits [15:0].	Invalid data is ignored, 1st valid data goes in LSB.
	O/P	None.	Indicates FIFO underflow.
IQ packing	I/P	Packing resynchronised to place sync data in bits [15:0].	Invalid data is ignored, 1st valid data goes in LSB.
	O/P	None.	Indicates FIFO underflow.

The effect of sync/valid on data packing is defined in the table below.

The effect of sync/valid on port multiplexing

“sync_valid_pin_function” configuration registers specify whether the valid pin is a sync or valid, and it whether it is active hign or low. When the valid pins is configured as a sync input and “reset-on-sync” is set the sync/valid pin enables the port multiplexing to be synchronous to incoming data packets. The port multiplexing will commence with the primary port when the first sync is received after start up or re-start. Data prior to the first sync is discarded. If bit 0 of the ADI sync word configuration register is set to 1 (“reset-on-sync”), the port used will revert to the primary port after each sync received, and dummy data will be sent to other ports if the sync is input at a point in the data stream destined for another port.

The only effect of input valid is that non-valid data is discarded. The port multiplexing cycles through valid data only.

For an output ADI the sync signals are taken from data from the sync ports (for output ADI ports 16 & 17 for channel 0 and 1 respectively). One sync bit is taken for each 16bit sample output, or 32bit sample in grouped mode. The sync bits do not effect the port multiplexing.

Support for Packet Data Structures

With Packet data structures a sync signal is often used as a timing reference to aid location of specific data fields within the packet. Typically specific Aes within the array will be programmed to process specific data fields within the packet, and must be able to uniquely locate them. This is supported using the sync port. On input, as each data sample is read from the data pins the value of the sync pin is sampled and placed in a bit of a register which corresponds with the data sample. The active value of sync (either high or low) is represented in the register as a 1, and the inactive value as a 0. Please refer to the picoTools User guide for further details of this register.

Start up

There are two start up options: either the ADI can start running at the same time as the rest of the chip, or the ADI data path can be disconnected internally until the rest of the chip is ready to source or sink data.

By setting the adi_restart bit high at configuration, the ADI will start running in response to a broadcast run command, but incoming external data will be discarded, or (for output) no GETs from the ports will be done and output data will be zeroed (and indicated as non valid if this option is chosen). Once the adi_restart bit is set low via the control port, the ADI begins normal operation. In valid mode input, the first valid data received will be PUT to the primary port, in sync mode input when reset on sync is selected the first data item which has an asserted sync will be PUT to the primary port otherwise the first data item read after startup will be put to the primary port. For output, GETs from the ports will commence with the sync port in sync mode, or the primary port otherwise, but until data arrives from the ports the output will be zeroed and indicated as non valid where appropriate.

The adi_restart register can be programmed by the host processor over the configuration bus, or by an AE using the control port as described in

As soon as ADI is started, the data rates must then continue to adhere to the conditions set out in section **Error! Reference source not found.**, until the picoArray is reset or the ADI is restarted.

Errors

FIFO errors are caused in the ADI when the hardware ports do not service the channels at sufficient rate and the FIFO either over or underflows.

In the case of an input channel if the FIFO overflows (written to when full) an error flag can be set; and the oldest four samples in the FIFO are lost.

In the case of an output channel the converse behaviour occurs. If the FIFO underflows (i.e. read from when empty) an error is indicated, and the FIFO read will repeat the last four samples of data alternately. If the ADI is configured to generate a data valid signal, this will also be de-asserted during these output samples.

FIFO errors can be read over the configuration bus. The error flags used to indicate underflow and overflow are the same, as a channel may only do one or the other dependent on its direction.

In normal operation the input channel FIFO can become empty or the output channel's FIFO can be full, the corresponding picoBus™ signal will be have no valid data or will stall respectively. Neither of these cases are a error case.

When □synchro the loss of data, it should be remembered that data is lost (due to overwriting) in the FIFO and hardware ports (which contain two pieces of data) are downstream of these, i.e. data is not lost from the oldest data in the ADI, but the oldest data in the FIFO.

Interfacing to a 16bit Data bus.

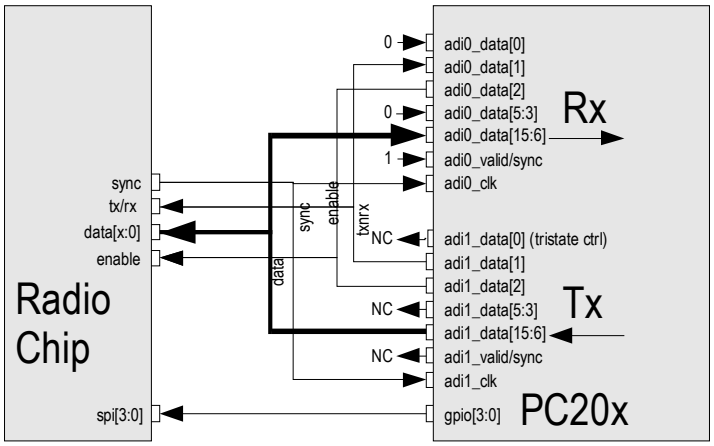
To allow the connection of bi-directional busses to the ADI interface is it possible to tristate a channel's data output pins (the valid pin is □synchrono) under the control of bit 0 of the data.
Additionally to data bit 0, the valid bit being de-asserted will also cause parts of the data output to be tristate if data dependent tristate is enabled. For this reason "Data dependent tristating" should not be used when the valid is used a sync bit.
This functionality is controlled by the upper bits of the "adi_channel_[01]_configuration" registers.

- Bit 15 – specifies the sense of the tri-state control of data bit 0
- Bit 14 – specifies that bits [15:4] of the data will be data dependent tri-stated.
- Bit 13 – specifies that bits [3:0] of the data will be data dependent tri-stated.

When the tri-state is active the ADI channel will perform all of its functions as normal except that data will not be driven from the specified pins.

The aim of this functionality is the direction of the data on the bi-directional bus can be controlled synchronously to the strobe. The tri-stating is controlled in two sections to allow, if needed, the tri-state control and other control wires to be driven out to external devices while the tri-state of other pins is in action. Figure 46 shows the typical Bi-directional interface connections.

Figure 38: Interfacing to 16bit Radio IC's



Interfacing to Multiplexed Radio processors.

In this configuration the Rx ADI always latches data on SYNC and so latches all of the RX and TX data, as well as the ENABLE and TX/RX control signals. From this stream of control and data the correct RX data can be extracted in software with the picoArray.

Data may be latched on the rising edge of SYNC, falling edge of SYNC, or both edges of SYNC. This is programmable in the PC20x devices.

In a similar way to the RX ADI continually latching data, the TX ADI always drives data on SYNC edges. So data must be provided for every edge of SYNC. ENABLE and TX/RX are merely 'extra' data bits as they are derived from the unused data bits. This means they have the same timing relationship to the SYNC clock as the data bits themselves. The state and polarity of these control signals can be determined in software by the picoArray. In this way ENABLE and TX/RX can be accurately controlled to the SYNC clock.

Since the picoArray can change the control signals of ENABLE and TX/RX signals this arrangement allows the ADI to trigger the ENABLE some fixed number of samples before the RX or TX data is available. Similarly the ENABLE can be triggered some fixed number of samples before the TX or RX frame is complete. This can be used to compensate for pipeline delays in the Radio Chip itself. This number is variable but must remain fixed for a particular configuration of Radio Chip. The ENABLE signal may either be pulsed (high or low) for some integral number of SYNC periods or it may be remain at a level for the duration of a TX or RX burst. The same is true for the TX/RX. Since these signals are derived from unused data bits their function and label is somewhat arbitrary; this allows alternative signal timing and functionality to be defined so long as it corresponds to

The bidirectional data bus is dealt with by the TX ADI's ability to send a tri-state signal on the LSB that stops the top 12 bits from driving. The change over of driving of the bus is controlled by the TX ADI. It drives two signals, through adi1_data[0] it controls its own interface (tri-state control) and through adi1_data[1] (TX/RX) it controls the tri-state of the Radio Chip. By using two separate signals for this all contention problems can be removed by allowing multiple SYNC periods to change over if needed.

4.7.1 ADI Timing parameters

Figure 39: ADI input / output timing for negative edge strobe Single Data Rate (SDR)

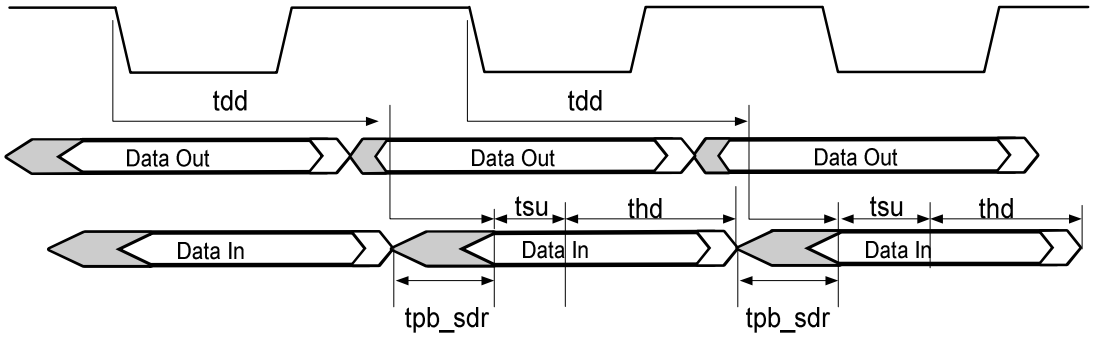


Figure 40: ADI input / output timing diagram for Double data rate (DDR)

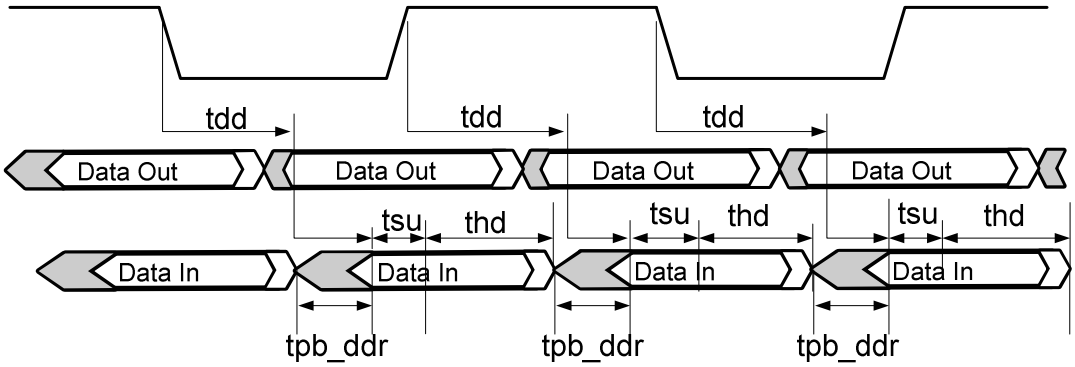


Table 33: ADI input / output timing for positive edge strobe Single Data Rate (SDR) & (DDR)

Parameter	Description	Min	Max	Units
tdd	Data out Delay	2.13	5.9	ns
tsu	Setup time	1.9		ns
thd	Hold Time	0.4		ns
tpd_ddr	Time delay caused by PCB DDR	1	1.75	ns
tpd_sdr	Time delay caused by PCB SDR	1	1.75	ns

Note 1: Input / Output data latched by either edges of the clock (here using negedge)

Note 2: Maximum ADI Clock for SDR mode is 160MHz.

Note 3: Maximum ADI Clock for DDR mode is 80MHz.

Note 4: The adi_clock_in should maintain a Marked Space Ratio of 60:40 or 40:60

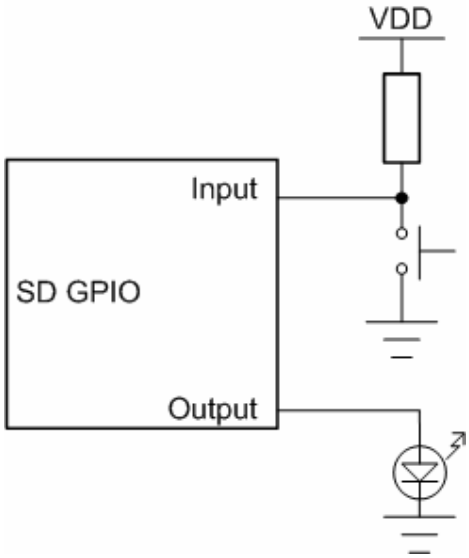
4.8 SD (Sigma-Delta) GPIO – PicoArray

The PC202 has 32 SD GPIO that can be programmed to perform several tasks including I²C. The initial status after RESET is Tristate.

Digital Operation

The SD GPIO sub-block can be used to sample pins configured as inputs and drive pins configured as outputs with a digital value. A typical example is shown in Figure 41.

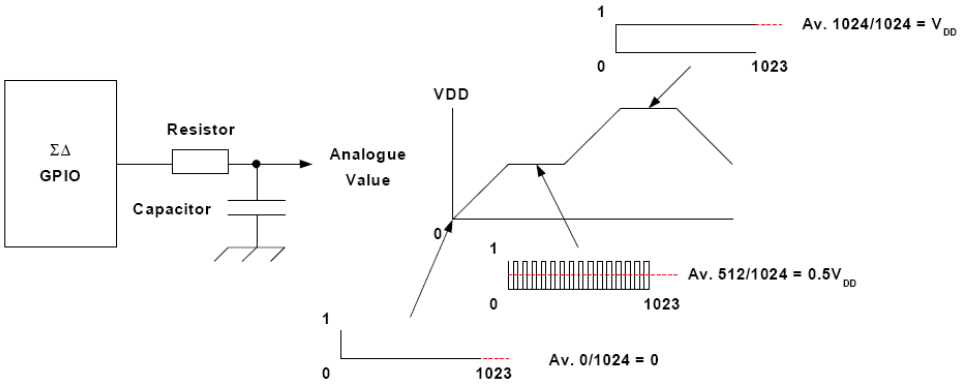
Figure 41: Typical Use Of SD GPIO Sub-Block



Analogue Operation

What differentiates the SD GPIO block from an ordinary GPIO block is the ability to drive analogue values on a pin. It does this by driving a ratio of high values to low values on the pin such that over the integration period the average DC voltage is that required.

Figure 42: The SD GPIO Sub-Block To Drive An Analogue Value



4.9 JTAG – (picoArray)

JTAG Signal Handling

The JTAG Interface has open-collector outputs requiring pull-ups on all lines with the exception of the /tmod and /trst signals, which should use 10kΩ pull-downs.

A small-value current-limiting series resistor, of value 4.7kΩ, should be included in the connection from the board +vddigital (either +1.8V or +3.3V).

Figure 43: JTAG interface timing diagram

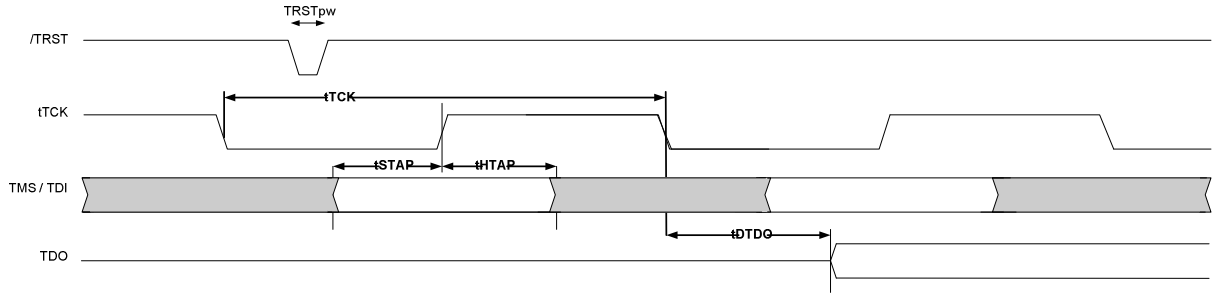


Table 34: JTAG interface timing parameters

Parameter	Description	Min	Max	Units
TRSTpw	TRST pulse width	25		ns
tTCK	JTAG TCK Period	100		ns
tSTAP	TDI, TMS Setup before T_{TCK} High	10		ns
tHTAP	TDI, TMS HOLD After T_{TCK} High	50		ns
tTDTDO	TDO Delay from T_{TCK} Low	50		ns

4.9.1 Instruction Register

The Instruction Register has 5 bits. The allocation of instruction codes is shown in Table 35.

Table 35: JTAG Instruction Codes

Code	Name	Status	Function
00000	EXTEST	Public	Allows testing from boundary-scan cells to the device pins
00001	SAMPLE/PRELOAD	Public	Allows capturing of data to/from the device pins, or preloading of the boundary-scan chain
00010	INTEST	Public	Allows testing from boundary-scan cells to the device core
00011 11011		Private	
11100	HIGHZ	Public	Puts all outputs into high impedance.
11101	CLAMP	Public	Switches pin control to the boundary scan cells
11110	IDCODE	Public	Selects the ID code register
11111	BYPASS	Public	Selects the bypass register

NOTE: The Instruction register loads the value 00001 (SAMPLE/PRELOAD) in the Capture-IR state.

4.9.2 Device ID Register

The Device ID Register contents are shown in Table 36.

Table 36: Device ID Register

Version	Part Number	Manufacturer	1
vvvv (Note 1)	0000 0000 0001 0000	0011 1010 100	1

Note 1: This is a 4-bit version number, which will change if there are functional revisions to the device. On the first production version of the PC202, this field is all zeros.

picoChip's JEDEC manufacturer ID is 0x54, bank 4. The DeviceID Register contents do not change in the Capture-DR state.

4.9.3 Boundary Scan Cell Control Bits

All the signal pins of the PC202 (except the TAP Controller pins themselves) are provided with full control/observe cells, including all clock inputs. Table 37 shows how the control bits are arranged for each of the pin types. Each pin can have up to three control bits. In the table, the first control bit listed is the one nearest TDO, i.e. this bit comes out first during a Shift-DR operation.

Table 37: BS Cell control bits

Pin type (Note 2)	Control bits	Control bit functions
Input	1	Input
Open Drain	1	Output
OD Bidirectional	2	Output, Input
3-state Outputs	2	Enable, Output (Note 1)
Bidirectional	3	Enable, Output, Input (Note 1)

Note 1: The polarity of the Enable bit is false, i.e. logic 1 causes the output to float.

Note 2: The HIGHZ instruction causes all boundary scan outputs to float.

4.9.4 Non-JTAG implementations

If the JTAG connections to the PC202 are not required, then tck, tmod and trst should have external 10kΩ pull downs to ground. TDI and TMS have internal pull-ups, and can be left unconnected. This will ensure that the Boundary Scan chain is inactive.

4.10 Ethernet 10/100 Interface

4.10.1 Overview

The Ethernet Mac provides support the following modes of operation.

Ethernet Mac 10/100

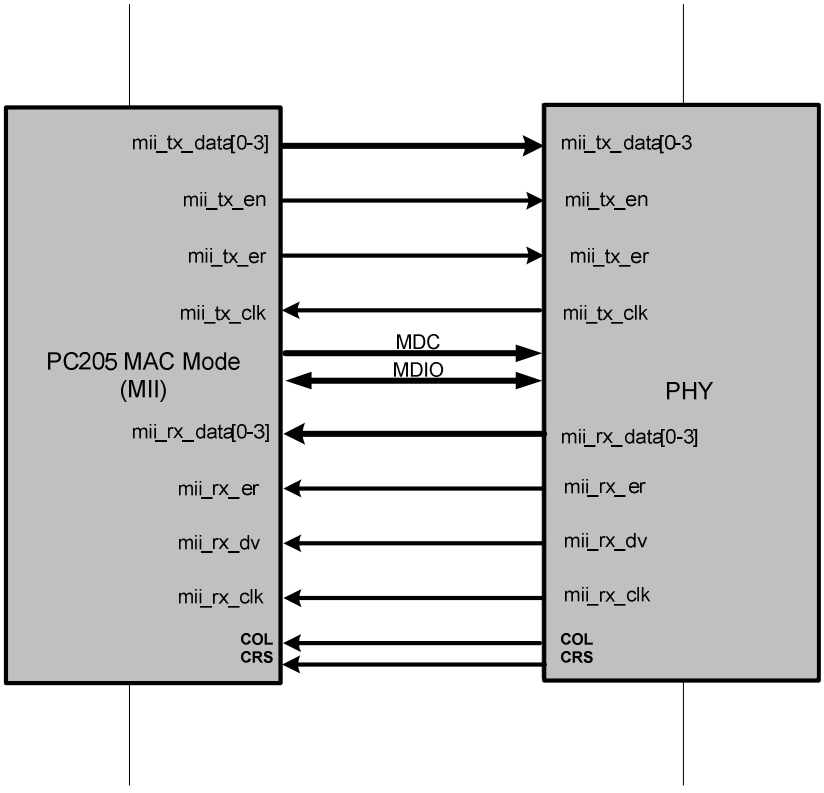
Reduced MII 10/100

Reverse MII 10/100

4.10.2 Ethernet 10/100 Interface

The PC202 Provides a standard Ethernet 10/100 Interface for the connection directly to industry standard physical layer Ethernet devices. Configuration and data formatting information can be found within the PC20X programmers guide. Figure 44 shows the connection from the PC202 to a Ethernet PHY

Figure 44: MII Block Diagram



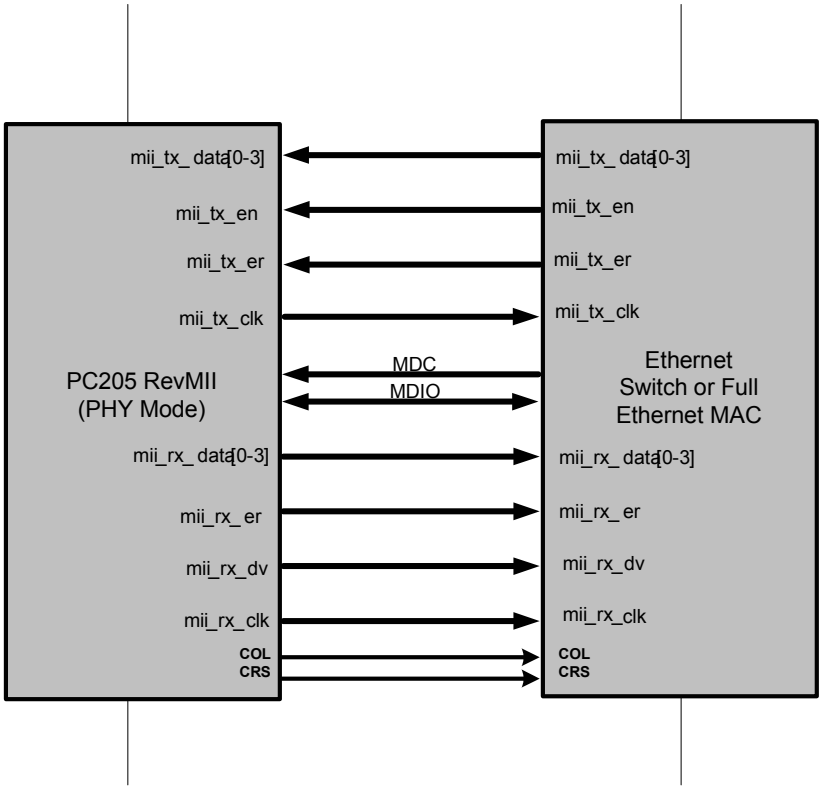
4.10.3 Ethernet 10/100 Interface (Reverse Mode)

Reverse MII, also known as PHY mode MII, is a means of connecting two MII interfaces back to back but still retaining the functionality required to manage the interfaces from software.

A block diagram showing the interface is shown in Figure 44. The interface consists of a series of multiplexers controlled from the serial management block. As there are effectively two masters to the control of this interface then some degree of arbitration and priority needs to be applied to ensure deadlock situations do not arise or can be resolved simply.

Reverse MII mode is selected by tying the mii_rev_en pin high. If the mii_rev_en pin is tied low, MII mode is selected.

Figure 45: MII Reverse Block Diagram



4.10.4 Ethernet 10/100 Interface (Reduced Mode)

The PC202 supports Reduced MII mode (RMII). This cuts the number of required signals from 16 to 7. There is a Common 50Mhz TX and RX ref clock, only a 2 bit wide tx and rx data path, no TX_er violation on the wire, No collision signal and (Carrier_sense & Receive_data_valid) are combined into CRS_DV. In RMII mode mii_rx_clk is used for the 50Mhz Reference clock.

The Following signal is used to configure the Reduced MII Speed

Speed Select Pad **E2** 10Mbs – Pull Down / 100Mbs- Pull Up mii_speed_sel is only used for RMII configuration.

Figure 46 shows the connection between the PC202 and Ethernet Phy/switch in RMII mode.

Note: Reduced Reversed MII mode is not supported by the PC202

Figure 46: MII Reduced Block Diagram

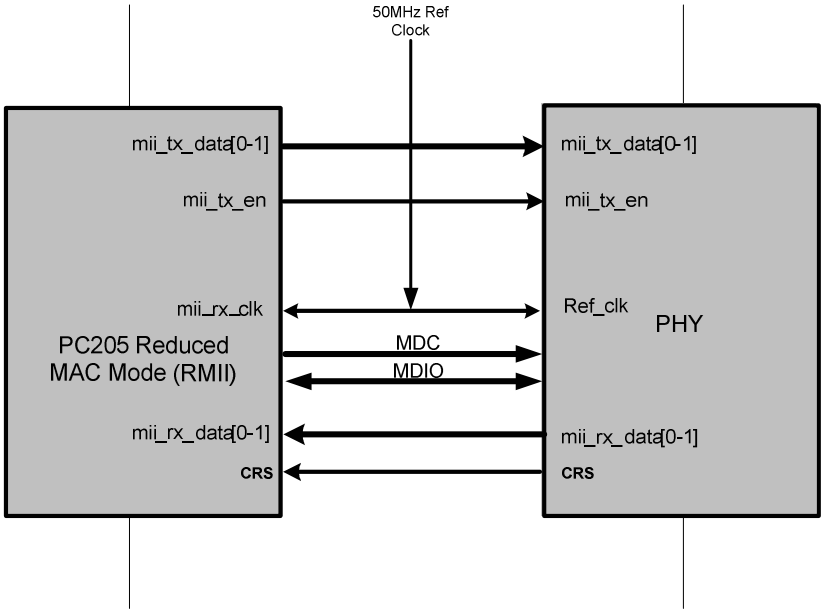


Figure 47: MII interface timing diagram

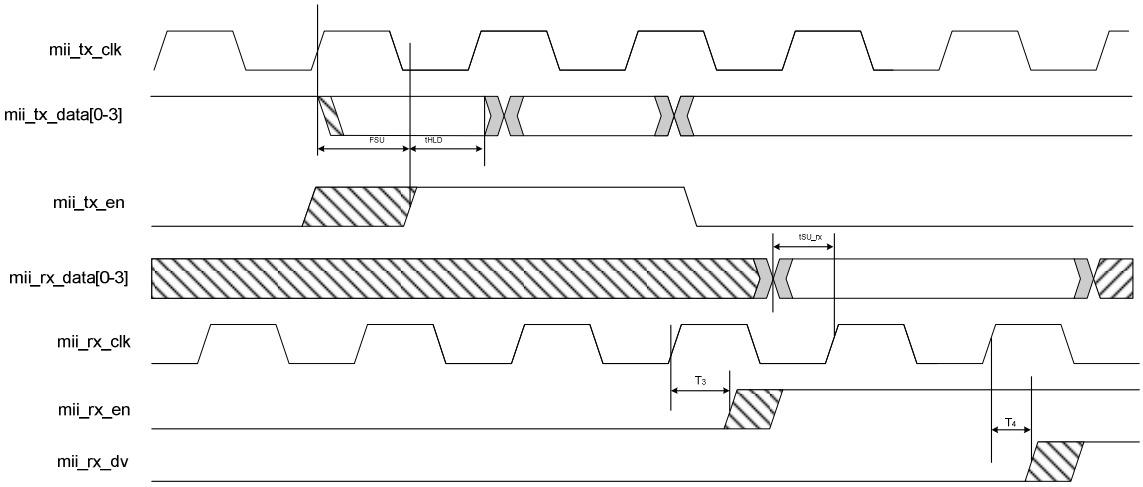


Table 38: MII interface timing

Parameter	Description	Min	Max	Units
tCLK	Tclk Time period	40	40	ns
tSU	tx_data[0-3], TX Set up time Delay from tx_clk High	3.5		ns
tHLD	tx_data[0-3], TX Hold after tx_clk High		35	ns
tSU_rx	tx_data[0-3], TX Set up time before rx_clk High	9.75		ns
T3	rx_data[0-3], rx_en Hold After rx_clk High	9.75		ns
T4	rx_data[0-3], rx_dv Hold After rx_clk High	4		ns

Figure 48: Reverse MII interface timing diagram

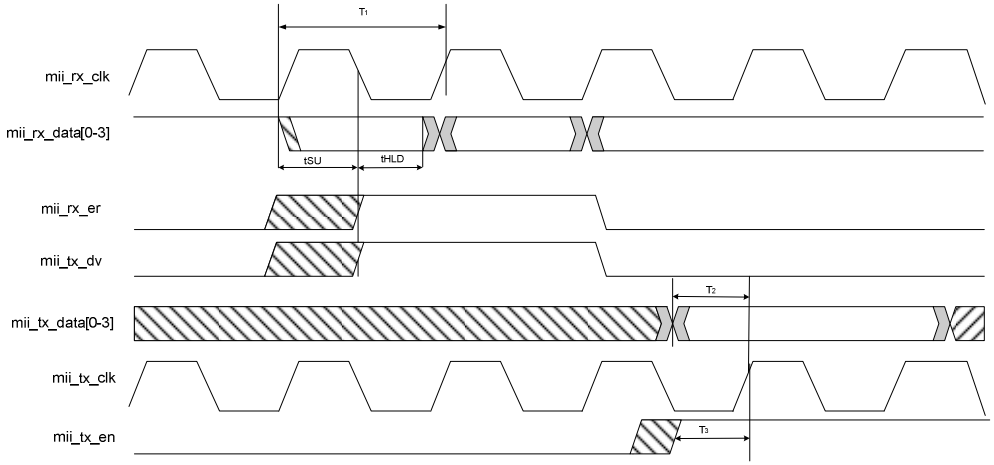


Table 39: Reverse MII interface timing

Parameter	Description	Min	Max	Units
tCLK	Tclk Time period	40	40	ns
tSU	Set up time after rising edge of rx clock	4.5		ns
tHLD	Hold time after rising edge of rx clock		35	ns
T2	Tx_data set up before rising edge	6.7		ns
T3	Te _en set up before rising edge	7.1		ns

Figure 49: Reduced MII interface timing diagram

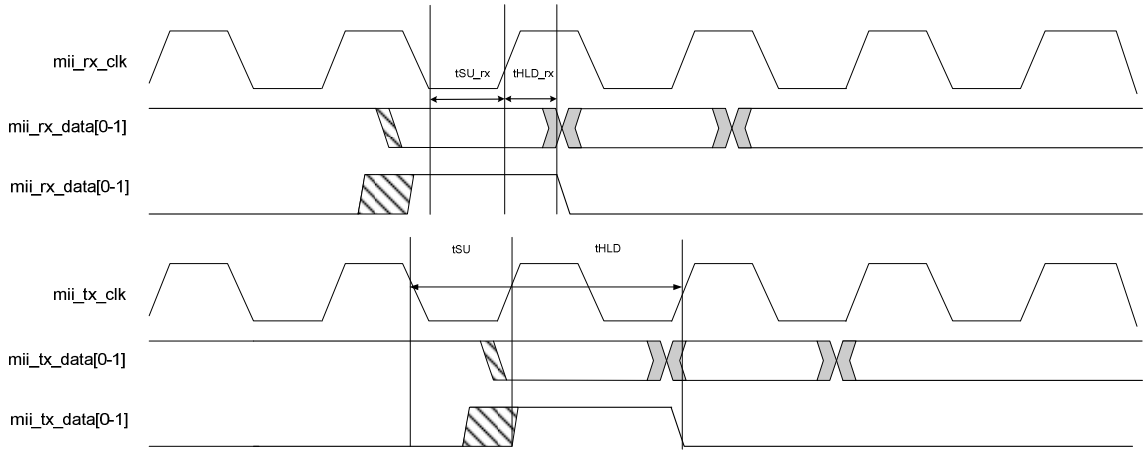


Table 40: Reduced MII interface timing

Parameter	Description	Min	Max	Units
tCLK	RX/TX clock period	20	20	ns
tSU_rx	rx_data[0-1] rx_dv, RX Set up time Delay from rx_clk High	3.3		ns
tHLD_rx	rx_data[0-1], RX Hold after rx_clk High		13	ns
tSU	tx_data[0-1] tx_en, TX Set up time Delay from tx_clk High	9.4		ns
tHLD	tx_data[0-1], TX Hold after tx_clk High		10	ns

4.11 The ARM925EJ Sub-system

The ARM-Sub-System contains an Ethernet MAC, dynamic and static memory controllers and an assortment of APB (ARM Peripheral Bus) peripherals such as GPIO (General Purpose I/O), UART, WDT (Watchdog Timer) and RTC (Real Time Clock). The dynamic memory system is accessed via a MEM-IF (Memory Interface) shared with the picoArray sub-system data streams. The Proc-IF (Processor Interface) provides the ARM-Sub-System with access to the picoArray's configuration bus and picoBus systems and an additional GPR (General Purpose Registers) and DMA interface provides extra-bandwidth between the picoArray-Sub-System and the ARM-Sub-System. The Proc-IF also provides access to the dynamic memory to an external processor attached to the EBI (Extension Bus Interface). This allows data to be passed between the external processor and the ARM926EJ.

Static and non-volatile memory can be accessed by the ARM-Sub-System via the EBI to provide configuration data and allow persistent storage.

4.11.1 Guide to ARM® Subsystem Programming Documents

Programming and operation of various elements of the ARM® Subsystem are contained in **PC202/5 ARM Subsystem & PC203 Programmer's Guide** and its appendices A-N which are contained in seprate documents. Below is a list of the contents of each appendix (A-N).

A	Memory Interface Configuration
B	AHB2Pico Interface Configuration
C	Direct Memory Access Controller (DMAC) Configuration
D	Vectored Interrupt Controller (VIC) Configuration
E	Ethernet MAC Configuration
F	General-Purpose Input/Output (GPIO) Configuration
G	Real-Time Clock (RTC) Configuration
H	Remap/Reset Register Configuration
I	Timers Configuration
J	Universal Asynchronous Receiver/Transmitter (UART) Configuration
K	Watchdog Timer Configuration
L	Processor Interface Access & Configuration
M	Memory Interface Access Controller Configuration
N	Sigma-Delta General-Purpose Input/Output Configuration

4.11.2 GPIO – (ARM®)

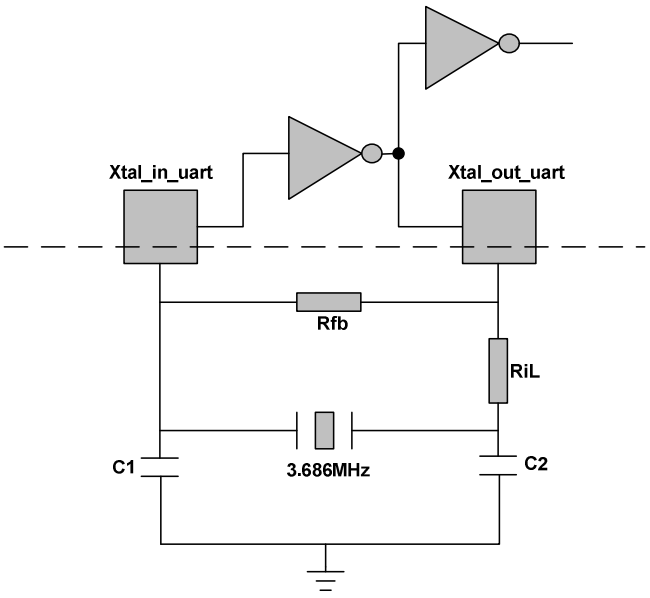
The arm_gpio[7:0] sub-block can be used to sample external pins configured as inputs and drive pins configured as outputs with a digital value. The GPIO be configured to generate interrupts.

4.11.3 UART – RS-232

There are two Universal Asynchronous Receiver/Transmitter (UART) peripherals on the APB, providing low bandwidth serial communications interfaces to the ARM926EJ. They are modeled on the industry-standard 16550 with registers to control the character length, baud rate, parity generation/checking and interrupt generation. There is only one interrupt output, although there are a number of prioritized interrupt types that can be selected using the control registers. Neither UART supports hardware flow control.

Internal 32-byte FIFOs can be used to buffer, transmit and receive data. The baud rate is a function of the serial clock frequency divided by 16 times the baud rate divisor. Both UARTs share the same serial clock.

Figure 50: UART Xtal Connection



4.11.4 Timers / RTC

Watchdog Timer

The Watchdog Timer is a counter that counts down from a programmed value to zero. When zero is reached it can either reset the system or generate an interrupt. If the interrupt is not cleared before a secondary counter reaches zero, the system is reset. The timer provides a safety feature to enable the system to return to a known state. Under normal operation, the timer will be reset by an interrupt service routine or a slow loop running in the application code.

General purpose Timer

There are four individually programmable 32-bit timers. They count down from a programmed value and generate an interrupt when they reach zero.

Real Time Clock

The Real-Time Clock is an incrementing counter for time keeping, with a programmable comparator for interrupt generation. Application software must determine the relationship between the counter value and actual time.

Please refer to the PC20X programmers guide for further details.

4.11.5 JTAG – (ARM)

Figure 51: JTAG interface timing diagram

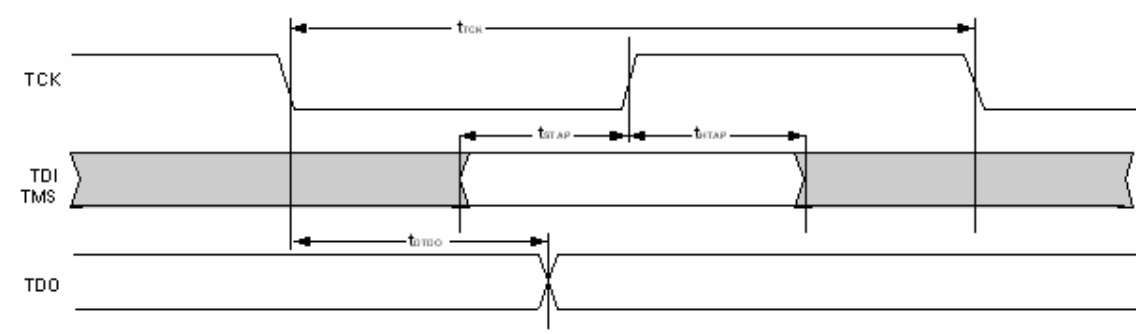


Table 41: JTAG interface timing parameters

Parameter	Description	Min	Max	Units
T_{TCK}	JTAG TCK Period	tbd	tbd	ns
T_{STAP}	TDI, TMS Setup before T_{TCK} High	tbd	tbd	ns
T_{HTAP}	TDI, TMS HOLD After T_{TCK} High	tbd	tbd	ns
T_{TDTDO}	TDO Delay from T_{TCK} Low	tbd	tbd	ns

Table 42: ARM JTAG Device ID Register

JTAG Number
0x0792_63A9

5 Pin Descriptions

Table 43 : ADI Interface pin descriptions (1.8vI/O)

Signal	Type	No.	Function
pdb0_dat[4]	Bidir	AA1	ADI Data Bus Channel 0
pdb0_dat[0]	Bidir	AA2	ADI Data Bus Channel 0
pdb0_dat[15]	Bidir	U2	ADI Data Bus Channel 0
pdb0_dat[13]	Bidir	U3	ADI Data Bus Channel 0
pdb0_dat[14]	Bidir	V2	ADI Data Bus Channel 0
pdb0_dat[11]	Bidir	V3	ADI Data Bus Channel 0
pdb0_dat[8]	Bidir	V4	ADI Data Bus Channel 0
pdb0_dat[9]	Bidir	V5	ADI Data Bus Channel 0
pdb0_dat[12]	Bidir	W1	ADI Data Bus Channel 0
pdb0_dat[10]	Bidir	W2	ADI Data Bus Channel 0
pdb0_dat[6]	Bidir	W3	ADI Data Bus Channel 0
pdb0_dat[2]	Bidir	W4	ADI Data Bus Channel 0
pdb0_dat[3]	Bidir	W5	ADI Data Bus Channel 0
pdb0_dat[7]	Bidir	Y1	ADI Data Bus Channel 0
pdb0_dat[5]	Bidir	Y2	ADI Data Bus Channel 0
pdb0_dat[1]	Bidir	Y3	ADI Data Bus Channel 0
pdb1_dat[0]	Bidir	AF5	ADI Data Bus Channel 1
pdb1_dat[13]	Bidir	AA5	ADI Data Bus Channel 1
pdb1_dat[14]	Bidir	AB4	ADI Data Bus Channel 1
pdb1_dat[6]	Bidir	AB5	ADI Data Bus Channel 1
pdb1_dat[15]	Bidir	AC3	ADI Data Bus Channel 1
pdb1_dat[10]	Bidir	AC4	ADI Data Bus Channel 1
pdb1_dat[5]	Bidir	AC5	ADI Data Bus Channel 1
pdb1_dat[12]	Bidir	AD3	ADI Data Bus Channel 1
pdb1_dat[8]	Bidir	AD4	ADI Data Bus Channel 1
pdb1_dat[4]	Bidir	AD5	ADI Data Bus Channel 1
pdb1_dat[11]	Bidir	AE2	ADI Data Bus Channel 1
pdb1_dat[9]	Bidir	AE3	ADI Data Bus Channel 1
pdb1_dat[7]	Bidir	AE4	ADI Data Bus Channel 1
pdb1_dat[1]	Bidir	AE5	ADI Data Bus Channel 1
pdb1_dat[3]	Bidir	AF3	ADI Data Bus Channel 1
pdb1_dat[2]	Bidir	AF4	ADI Data Bus Channel 1
pdb1_in0	In	AC2	ADI Channel 0 Input strobe
pdb1_in1	In	AB3	ADI Channel 1 input strobe
pdb1_val	Bidir	AD1	ADI Channel 1 Valid/sync in/out.
Pdb0_val	Bidir	Y5	ADI Channel 0 Valid/sync in/out.

Table 44: Core / GND pin descriptions

Signal	Type	No.	Function
gnd	Power	A18	Core Power / Grounds
gnd	Power	A25	Core Power / Grounds
gnd	Power	A9	Core Power / Grounds
gnd	Power	AE1	Core Power / Grounds
gnd	Power	AE26	Core Power / Grounds
gnd	Power	AF18	Core Power / Grounds
gnd	Power	AF2	Core Power / Grounds
gnd	Power	AF25	Core Power / Grounds
gnd	Power	AF9	Core Power / Grounds
gnd	Power	B1	Core Power / Grounds
gnd	Power	B26	Core Power / Grounds
pll_vss	Power	D16	Core Power / Grounds
pll_vdd	Power	E15	Core Power / Grounds
pll_avdd	Power	E16	Core Power / Grounds
pll_avss	Power	E17	Core Power / Grounds
vddcore	Power	G10	Core Power / Grounds
vddcore	Power	G11	Core Power / Grounds
vddcore	Power	G12	Core Power / Grounds
vddcore	Power	G13	Core Power / Grounds
vddcore	Power	G14	Core Power / Grounds
vddcore	Power	G15	Core Power / Grounds
vddcore	Power	G16	Core Power / Grounds
vddcore	Power	G17	Core Power / Grounds
vddcore	Power	G18	Core Power / Grounds
vddcore	Power	G19	Core Power / Grounds
vddcore	Power	G20	Core Power / Grounds
vddcore	Power	G7	Core Power / Grounds
vddcore	Power	G8	Core Power / Grounds
vddcore	Power	G9	Core Power / Grounds
vddcore	Power	H10	Core Power / Grounds
vddcore	Power	H11	Core Power / Grounds
vddcore	Power	H12	Core Power / Grounds
vddcore	Power	H13	Core Power / Grounds
vddcore	Power	H14	Core Power / Grounds
vddcore	Power	H15	Core Power / Grounds
vddcore	Power	H16	Core Power / Grounds
vddcore	Power	H17	Core Power / Grounds
vddcore	Power	H18	Core Power / Grounds
vddcore	Power	H19	Core Power / Grounds
vddcore	Power	H20	Core Power / Grounds
vddcore	Power	H7	Core Power / Grounds
vddcore	Power	H8	Core Power / Grounds
vddcore	Power	H9	Core Power / Grounds
gnd	Power	J1	Core Power / Grounds
gnd	Power	J10	Core Power / Grounds
gnd	Power	J11	Core Power / Grounds
gnd	Power	J12	Core Power / Grounds
gnd	Power	J13	Core Power / Grounds
gnd	Power	J14	Core Power / Grounds
gnd	Power	J15	Core Power / Grounds
gnd	Power	J16	Core Power / Grounds
gnd	Power	J17	Core Power / Grounds
gnd	Power	J18	Core Power / Grounds
vddcore	Power	J19	Core Power / Grounds
vddcore	Power	J20	Core Power / Grounds
gnd	Power	J26	Core Power / Grounds
vddcore	Power	J7	Core Power / Grounds
vddcore	Power	J8	Core Power / Grounds
gnd	Power	J9	Core Power / Grounds
gnd	Power	K10	Core Power / Grounds
gnd	Power	K11	Core Power / Grounds
gnd	Power	K12	Core Power / Grounds
gnd	Power	K13	Core Power / Grounds
gnd	Power	K14	Core Power / Grounds

Signal	Type	No.	Function
gnd	Power	K15	Core Power / Grounds
gnd	Power	K16	Core Power / Grounds
gnd	Power	K17	Core Power / Grounds
gnd	Power	K18	Core Power / Grounds
vddcore	Power	K19	Core Power / Grounds
vddcore	Power	K20	Core Power / Grounds
vddcore	Power	K7	Core Power / Grounds
vddcore	Power	K8	Core Power / Grounds
gnd	Power	K9	Core Power / Grounds
gnd	Power	L10	Core Power / Grounds
gnd	Power	L11	Core Power / Grounds
gnd	Power	L12	Core Power / Grounds
gnd	Power	L13	Core Power / Grounds
gnd	Power	L14	Core Power / Grounds
gnd	Power	L15	Core Power / Grounds
gnd	Power	L16	Core Power / Grounds
gnd	Power	L17	Core Power / Grounds
gnd	Power	L18	Core Power / Grounds
vddcore	Power	L19	Core Power / Grounds
vddcore	Power	L20	Core Power / Grounds
vddcore	Power	L7	Core Power / Grounds
vddcore	Power	L8	Core Power / Grounds
gnd	Power	L9	Core Power / Grounds
gnd	Power	M10	Core Power / Grounds
gnd	Power	M11	Core Power / Grounds
gnd	Power	M12	Core Power / Grounds
gnd	Power	M13	Core Power / Grounds
gnd	Power	M14	Core Power / Grounds
gnd	Power	M15	Core Power / Grounds
gnd	Power	M16	Core Power / Grounds
gnd	Power	M17	Core Power / Grounds
gnd	Power	M18	Core Power / Grounds
vddcore	Power	M19	Core Power / Grounds
vddcore	Power	M20	Core Power / Grounds

Signal	Type	No.	Function
vddcore	Power	M7	Core Power / Grounds
vddcore	Power	M8	Core Power / Grounds
gnd	Power	M9	Core Power / Grounds
gnd	Power	N10	Core Power / Grounds
gnd	Power	N11	Core Power / Grounds
gnd	Power	N12	Core Power / Grounds
gnd	Power	N13	Core Power / Grounds
gnd	Power	N14	Core Power / Grounds
gnd	Power	N15	Core Power / Grounds
gnd	Power	N16	Core Power / Grounds
gnd	Power	N17	Core Power / Grounds
gnd	Power	N18	Core Power / Grounds
vddcore	Power	N19	Core Power / Grounds
vddcore	Power	N20	Core Power / Grounds
vddcore	Power	N7	Core Power / Grounds
vddcore	Power	N8	Core Power / Grounds
gnd	Power	N9	Core Power / Grounds
gnd	Power	P10	Core Power / Grounds
gnd	Power	P11	Core Power / Grounds
gnd	Power	P12	Core Power / Grounds
gnd	Power	P13	Core Power / Grounds
gnd	Power	P14	Core Power / Grounds
gnd	Power	P15	Core Power / Grounds
gnd	Power	P16	Core Power / Grounds
gnd	Power	P17	Core Power / Grounds
gnd	Power	P18	Core Power / Grounds
vddcore	Power	P19	Core Power / Grounds
vddcore	Power	P20	Core Power / Grounds
vddcore	Power	P7	Core Power / Grounds
vddcore	Power	P8	Core Power / Grounds
gnd	Power	P9	Core Power / Grounds
gnd	Power	R10	Core Power / Grounds
gnd	Power	R11	Core Power / Grounds
gnd	Power	R12	Core Power / Grounds

Signal	Type	No.	Function
gnd	Power	R13	Core Power / Grounds
gnd	Power	R14	Core Power / Grounds
gnd	Power	R15	Core Power / Grounds
gnd	Power	R16	Core Power / Grounds
gnd	Power	R17	Core Power / Grounds
gnd	Power	R18	Core Power / Grounds
vddcore	Power	R19	Core Power / Grounds
vddcore	Power	R20	Core Power / Grounds
vddcore	Power	R7	Core Power / Grounds
vddcore	Power	R8	Core Power / Grounds
gnd	Power	R9	Core Power / Grounds
gnd	Power	T10	Core Power / Grounds
gnd	Power	T11	Core Power / Grounds
gnd	Power	T12	Core Power / Grounds
gnd	Power	T13	Core Power / Grounds
gnd	Power	T14	Core Power / Grounds
gnd	Power	T15	Core Power / Grounds
gnd	Power	T16	Core Power / Grounds
gnd	Power	T17	Core Power / Grounds
gnd	Power	T18	Core Power / Grounds
vddcore	Power	T19	Core Power / Grounds
vddcore	Power	T20	Core Power / Grounds
vddcore	Power	T7	Core Power / Grounds
vddcore	Power	T8	Core Power / Grounds
gnd	Power	T9	Core Power / Grounds
gnd	Power	U10	Core Power / Grounds
gnd	Power	U11	Core Power / Grounds
gnd	Power	U12	Core Power / Grounds
gnd	Power	U13	Core Power / Grounds
gnd	Power	U14	Core Power / Grounds
gnd	Power	U15	Core Power / Grounds
gnd	Power	U16	Core Power / Grounds
gnd	Power	U17	Core Power / Grounds
gnd	Power	U18	Core Power / Grounds

Signal	Type	No.	Function
vddcore	Power	U19	Core Power / Grounds
vddcore	Power	U20	Core Power / Grounds
gnd	Power	U23	Core Power / Grounds
vddcore	Power	U7	Core Power / Grounds
vddcore	Power	U8	Core Power / Grounds
gnd	Power	U9	Core Power / Grounds
gnd	Power	V1	Core Power / Grounds
gnd	Power	V10	Core Power / Grounds
gnd	Power	V11	Core Power / Grounds
gnd	Power	V12	Core Power / Grounds
gnd	Power	V13	Core Power / Grounds
gnd	Power	V14	Core Power / Grounds
gnd	Power	V15	Core Power / Grounds
gnd	Power	V16	Core Power / Grounds
gnd	Power	V17	Core Power / Grounds
gnd	Power	V18	Core Power / Grounds
vddcore	Power	V19	Core Power / Grounds
vddcore	Power	V20	Core Power / Grounds
gnd	Power	V26	Core Power / Grounds
vddcore	Power	V7	Core Power / Grounds
vddcore	Power	V8	Core Power / Grounds
gnd	Power	V9	Core Power / Grounds
vddcore	Power	W10	Core Power / Grounds
vddcore	Power	W11	Core Power / Grounds
vddcore	Power	W12	Core Power / Grounds
vddcore	Power	W13	Core Power / Grounds
vddcore	Power	W14	Core Power / Grounds
vddcore	Power	W15	Core Power / Grounds
vddcore	Power	W16	Core Power / Grounds
vddcore	Power	W17	Core Power / Grounds
vddcore	Power	W18	Core Power / Grounds
vddcore	Power	W19	Core Power / Grounds
vddcore	Power	W20	Core Power / Grounds
vddcore	Power	W7	Core Power / Grounds

Signal	Type	No.	Function
vddcore	Power	W8	Core Power / Grounds
vddcore	Power	W9	Core Power / Grounds
vddcore	Power	Y10	Core Power / Grounds
vddcore	Power	Y11	Core Power / Grounds
vddcore	Power	Y12	Core Power / Grounds
vddcore	Power	Y13	Core Power / Grounds
vddcore	Power	Y14	Core Power / Grounds
vddcore	Power	Y15	Core Power / Grounds
vddcore	Power	Y16	Core Power / Grounds
vddcore	Power	Y17	Core Power / Grounds
vddcore	Power	Y18	Core Power / Grounds
vddcore	Power	Y19	Core Power / Grounds
vddcore	Power	Y20	Core Power / Grounds
vddcore	Power	Y7	Core Power / Grounds
vddcore	Power	Y8	Core Power / Grounds
vddcore	Power	Y9	Core Power / Grounds

Table 45: External Bus Interface pin descriptions

Signal	Type	No.	Function
ebi_busy[2]	In	A20	Busy signal
ebi_busy[1]	In	B19	Busy signal
ebi_busy[0]	In	A21	Busy signal
ebi_byte_mode[1]	Out	C20	Byte addressable mode. Used to access sub words and bytes of 16 and 32 bit wide memory types. Ebi byte mode[3] relates to the 4th byte of a 32 bit word.
Ebi_byte_mode[0]	Out	C21	
ebi_byte_mode[2]	Out	D20	
ebi_byte_mode[3]	Out	E20	
ebi_dwidth[1]	In	D18	Config for External data bus width
ebi_dwidth[2]	In	D19	Config for External data bus width
ebi_dwidth[0]	In	E19	Config for External data bus width
ebi_dreq[0]	In	A17	DMA Request
ebi_dreq[1]	In	A19	DMA Request
ebi_dreq[2]	In	B18	DMA Request
ebi_dreq[3]	In	C17	DMA Request
/ebi_oe	Out	A22	External Bus read enable
/ebi_we	Out	B21	External Bus write enable
ebi_clk	Out	D15	External Bus Output Clock
ebi_addr[5]	Out	C22	External Bus Interface Address
ebi_addr[7]	Out	C23	External Bus Interface Address
ebi_addr[8]	Out	C24	External Bus Interface Address
ebi_addr[14]	Out	C25	External Bus Interface Address
ebi_addr[17]	Out	C26	External Bus Interface Address
ebi_addr[9]	Out	D22	External Bus Interface Address
ebi_addr[1]	Out	A23	External Bus Interface Address
ebi_addr[2]	Out	A24	External Bus Interface Address
ebi_addr[0]	Out	B22	External Bus Interface Address
ebi_addr[4]	Out	B23	External Bus Interface Address
ebi_addr[3]	Out	B24	External Bus Interface Address
ebi_addr[13]	Out	B25	External Bus Interface Address
ebi_addr[6]	Out	D21	External Bus Interface Address
ebi_addr[10]	Out	E22	External Bus Interface Address
ebi_addr[11]	Out	E21	External Bus Interface Address
ebi_addr[12]	Out	D23	External Bus Interface Address
ebi_addr[15]	Out	D24	External Bus Interface Address
ebi_addr[16]	Out	E23	External Bus Interface Address
ebi_addr[18]	Out	D25	External Bus Interface Address
ebi_addr[19]	Out	E24	External Bus Interface Address
ebi_addr[20]	Out	F23	External Bus Interface Address
ebi_addr[21]	Out	F24	External Bus Interface Address
ebi_addr[22]	Out	E25	External Bus Interface Address
ebi_addr[23]	Out	D26	External Bus Interface Address
ebi_addr[24]	Out	G24	External Bus Interface Address
ebi_addr[25]	Out	F25	External Bus Interface Address
ebi_addr[26]	Out	E26	External Bus Interface Address
ebi_addr[27]	Out	F26	External Bus Interface Address
ebi_addr[28]	Out	G25	External Bus Interface Address
ebi_data[10]	Bidir	A7	External Bus Interface Data
ebi_data[13]	Bidir	C8	External Bus Interface Data
ebi_data[14]	Bidir	B8	External Bus Interface Data
ebi_data[15]	Bidir	A8	External Bus Interface Data
ebi_data[18]	Bidir	C9	External Bus Interface Data
ebi_data[19]	Bidir	B9	External Bus Interface Data
ebi_data[22]	Bidir	C10	External Bus Interface Data
ebi_data[23]	Bidir	B10	External Bus Interface Data
ebi_data[24]	Bidir	A10	External Bus Interface Data
ebi_data[27]	Bidir	C11	External Bus Interface Data
ebi_data[28]	Bidir	B11	External Bus Interface Data
ebi_data[29]	Bidir	A11	External Bus Interface Data
ebi_data[3]	Bidir	C6	External Bus Interface Data
ebi_data[5]	Bidir	A5	External Bus Interface Data
ebi_data[6]	Bidir	B6	External Bus Interface Data
ebi_data[7]	Bidir	A6	External Bus Interface Data
ebi_data[8]	Bidir	C7	External Bus Interface Data
ebi_data[9]	Bidir	B7	External Bus Interface Data

Signal	Type	No.	Function
/ebi_decode[0]	Out	C12	External Chip Decode Area
/ebi_decode[1]	Out	B12	External Chip Decode Area
/ebi_decode[2]	Out	A12	External Chip Decode Area
/ebi_decode[3]	Out	E13	External Chip Decode Area
/ebi_decode[4]	Out	D13	External Chip Decode Area
/ebi_decode[5]	Out	C13	External Chip Decode Area
/ebi_decode[6]	Out	D14	External Chip Decode Area
/ebi_decode[7]	Out	B15	External Chip Decode Area
ebi_data[0]	Bidir	E7	External Bus Interface Data
ebi_data[1]	Bidir	E6	External Bus Interface Data
ebi_data[11]	Bidir	E8	External Bus Interface Data
ebi_data[12]	Bidir	D8	External Bus Interface Data
ebi_data[16]	Bidir	E9	External Bus Interface Data
ebi_data[17]	Bidir	D9	External Bus Interface Data
ebi_data[2]	Bidir	D6	External Bus Interface Data
ebi_data[20]	Bidir	E10	External Bus Interface Data
ebi_data[21]	Bidir	D10	External Bus Interface Data
ebi_data[25]	Bidir	E11	External Bus Interface Data
ebi_data[26]	Bidir	D11	External Bus Interface Data
ebi_data[30]	Bidir	E12	External Bus Interface Data
ebi_data[31]	Bidir	D12	External Bus Interface Data
ebi_data[4]	Bidir	D7	External Bus Interface Data
ebi_ready	In	C15	RDY signal
ebi_rp	Out	E18	NOR Flash Read protect signal
ebi_wp[0]	Out	C18	NOR Flash Write protect signal
ebi_wp[1]	Out	C19	NOR Flash Write protect signal
ebi_wp[2]	Out	B20	NOR Flash Write protect signal

Table 46: Microprocessor Interface (3.3v IO)

Signal	Type	No.	Function
/up_cs[0]	In	C12	Decode region 0. Access picoArray registers.
/up_cs[1]	In	B12	Decode region 1. Accesses picoArray DMA
/up_cs[2]	In	A12	Decode region 2. Access controller registers.
/up_oe	In	A22	Read Access
/up_we	In	B21	Write Access
up_adlo[0]	In	B22	Adlo [1:0]: Used by all services when in 16-bit or 8-bit mode. Bit [0] must be zero in 32-bit or 16 bit mode, bit[1] must be zero in 32-bit mode. Addresses have both a synchronous relationship with proc_if_clk and an □synchronous relationship with the data pins in 16-bit and 8 bit mode.
Up_adlo[1]	In	A23	
up_adlo[2]	In	A24	
up_adlo[3]	In	B24	
up_adlo[4]	In	B23	
up_adlo[5]	In	C22	
up_adlo[6]	In	D21	Adlo [6:2]: Used when addressing non-DMA services via CS [2] and CS[0], ignored when using CS[1] services.
up_adhi[0]	In	C23	Adhi [1:0]: Used when addressing DMA services via CS[1] ignored when using CS[0].
up_adhi[1]	In	C24	
proc_irq	Out	C16	Processor interface irq
proc_clk	In	C14	Proc_IF Clock Maximum Frequency 100Mhz***

***** proc_clk Must be pulled down** when the devices is not used in Slave mode and the device uses the EBI.

Signal	Type	No.	Function
up_mode[0]	In	C21	Up_mode[2:0] Select the Proc IF Mode. (Allow 16 Proc_IF_clk Cycles before and after any change. Up_mode [2:0] — 010: 32-bit Mode; 001: 16-bit Mode; 000: 8-bit Mode; 100: High to Low Edge; 000: Low to High Edge
up_mode[1]	In	C20	
up_mode[2]	In	D20	
up_data[0]	Bidir	E7	32 bit data bus, in 16-bit mode bits [31:16] and 8-bit mode bits [31:8] are unused and are undriven.
Up_data[1]	Bidir	E6	
up_data[10]	Bidir	A7	
up_data[11]	Bidir	E8	
up_data[12]	Bidir	D8	
up_data[13]	Bidir	C8	
up_data[14]	Bidir	B8	
up_data[15]	Bidir	A8	
up_data[16]	Bidir	E9	
up_data[17]	Bidir	D9	
up_data[18]	Bidir	C9	
up_data[19]	Bidir	B9	
up_data[2]	Bidir	D6	
up_data[20]	Bidir	E10	
up_data[21]	Bidir	D10	
up_data[22]	Bidir	C10	
up_data[23]	Bidir	B10	
up_data[24]	Bidir	A10	
up_data[25]	Bidir	E11	
up_data[26]	Bidir	D11	
up_data[27]	Bidir	C11	
up_data[28]	Bidir	B11	
up_data[29]	Bidir	A11	

Signal	Type	No.	Function
up_data[3]	Bidir	C6	32 bit data bus, in 16-bit mode bits [31:16] and 8-bit mode bits [31:8] are unused and are undriven.
Up_data[30]	Bidir	E12	
up_data[31]	Bidir	D12	
up_data[4]	Bidir	D7	
up_data[5]	Bidir	A5	
up_data[6]	Bidir	B6	
up_data[7]	Bidir	A6	
up_data[8]	Bidir	C7	
up_data[9]	Bidir	B7	
/up_dreq[0]	Out	A17	Processor interface DMA Request
/up_dreq[1]	Out	A19	Processor interface DMA Request
/up_dreq[2]	Out	B18	Processor interface DMA Request
/up_dreq[3]	Out	C17	Processor interface DMA Request

Table 47: Handling of Unused EBI interface pins in Slave Mode

Signal	Type	No.	Function
ebi_addr[28:9]	In		Pull Down
ebi_byte mode [3]	In		Pull Down
ebi_decode[7:3]	In		Pull Down
ebi_dwidth[2:0]	In		Pull Down
ebi_rp	Out		Do not connect
ebi_wp[2:0]	Out		Do not connect
ebi_busy[2:0]	In		Pull Down
ebi_ready	In		Pull Down
ebi_clk	Out		Do not connect

Table 48: GPIO pin descriptions

Signal	Type	No.	Function
arm_gpio[7]	Bidir	H1	ARM GPIO
arm_gpio[0]	Bidir	K2	ARM GPIO
arm_gpio[1]	Bidir	J2	ARM GPIO
arm_gpio[2]	Bidir	K3	ARM GPIO
arm_gpio[3]	Bidir	L5	ARM GPIO
arm_gpio[4]	Bidir	J5	ARM GPIO
arm_gpio[5]	Bidir	K5	ARM GPIO
arm_gpio[6]	Bidir	K4	ARM GPIO
sd_gpio[0]	Bidir	U4	Sigma Delta GPIO
sd_gpio[1]	Bidir	U5	Sigma Delta GPIO
sd_gpio[10]	Bidir	R4	Sigma Delta GPIO
sd_gpio[11]	Bidir	R5	Sigma Delta GPIO
sd_gpio[12]	Bidir	P1	Sigma Delta GPIO
sd_gpio[13]	Bidir	P2	Sigma Delta GPIO
sd_gpio[14]	Bidir	P3	Sigma Delta GPIO
sd_gpio[15]	Bidir	P4	Sigma Delta GPIO
sd_gpio[16]	Bidir	P5	Sigma Delta GPIO
sd_gpio[17]	Bidir	N1	Sigma Delta GPIO
sd_gpio[18]	Bidir	N2	Sigma Delta GPIO
sd_gpio[19]	Bidir	N3	Sigma Delta GPIO
sd_gpio[2]	Bidir	T1	Sigma Delta GPIO
sd_gpio[20]	Bidir	N4	Sigma Delta GPIO
sd_gpio[21]	Bidir	N5	Sigma Delta GPIO
sd_gpio[22]	Bidir	M1	Sigma Delta GPIO
sd_gpio[23]	Bidir	M2	Sigma Delta GPIO
sd_gpio[24]	Bidir	M3	Sigma Delta GPIO
sd_gpio[25]	Bidir	M4	Sigma Delta GPIO
sd_gpio[26]	Bidir	M5	Sigma Delta GPIO
sd_gpio[27]	Bidir	L1	Sigma Delta GPIO
sd_gpio[28]	Bidir	L2	Sigma Delta GPIO
sd_gpio[29]	Bidir	L3	Sigma Delta GPIO
sd_gpio[3]	Bidir	T2	Sigma Delta GPIO
sd_gpio[30]	Bidir	L4	Sigma Delta GPIO
sd_gpio[31]	Bidir	K1	Sigma Delta GPIO
sd_gpio[4]	Bidir	T3	Sigma Delta GPIO
sd_gpio[5]	Bidir	T4	Sigma Delta GPIO
sd_gpio[6]	Bidir	T5	Sigma Delta GPIO
sd_gpio[7]	Bidir	R1	Sigma Delta GPIO
sd_gpio[8]	Bidir	R2	Sigma Delta GPIO
sd_gpio[9]	Bidir	R3	Sigma Delta GPIO

Table 49: I/O Power pin descriptions

Signal	Type	No.	Function
vddadi	Power	AA10	I/O Power
vddadi	Power	AA11	I/O Power
vddadi	Power	AA12	I/O Power
vddadi	Power	AA13	I/O Power
vddadi	Power	AA14	I/O Power
vddadi	Power	AA15	I/O Power
vddadi	Power	AA16	I/O Power
vddadi	Power	AA17	I/O Power
vddadi	Power	AA18	I/O Power
vddadi	Power	AA19	I/O Power
vddadi	Power	AA20	I/O Power
vddmem	Power	AA21	I/O Power
vddadi	Power	AA6	I/O Power
vddadi	Power	AA7	I/O Power
vddadi	Power	AA8	I/O Power
vddadi	Power	AA9	I/O Power
vddio	Power	F10	I/O Power
vddio	Power	F11	I/O Power
vddio	Power	F12	I/O Power
vddio	Power	F13	I/O Power
vddio	Power	F14	I/O Power
vddio	Power	F15	I/O Power
vddio	Power	F16	I/O Power
vddio	Power	F17	I/O Power
vddio	Power	F18	I/O Power
vddio	Power	F19	I/O Power
vddio	Power	F20	I/O Power
vddio	Power	F21	I/O Power
vddio	Power	F6	I/O Power
vddio	Power	F7	I/O Power
vddio	Power	F8	I/O Power
vddio	Power	F9	I/O Power
vddio	Power	G21	I/O Power
vddio	Power	G6	I/O Power
vddio	Power	H21	I/O Power
vddio	Power	H6	I/O Power
vddio	Power	J21	I/O Power
vddio	Power	J6	I/O Power
vddmem	Power	K21	I/O Power
vddio	Power	K6	I/O Power
vddmem	Power	L21	I/O Power
vddio	Power	L6	I/O Power
vddmem	Power	M21	I/O Power
vddio	Power	M6	I/O Power
vddmem	Power	N21	I/O Power
vddio	Power	N6	I/O Power
vddmem	Power	P21	I/O Power
vddio	Power	P6	I/O Power
vddmem	Power	R21	I/O Power
vddio	Power	R6	I/O Power
vddmem	Power	T21	I/O Power
vddadi	Power	T6	I/O Power
vddmem	Power	U21	I/O Power
vddadi	Power	U6	I/O Power
vddmem	Power	V21	I/O Power
vddadi	Power	V6	I/O Power
vddmem	Power	W21	I/O Power
vddadi	Power	W6	I/O Power
vddmem	Power	Y21	I/O Power
vddadi	Power	Y6	I/O Power

Table 50: MII & UART pin descriptions

Signal	Type	No.	Function
mii_rx_er	Bidir	A3	MII
mii_addr[0]	In	G3	MII
mii_addr[1]	In	H4	MII
mii_addr[2]	In	G4	MII
mii_addr[3]	In	C2	MII
mii_addr[4]	In	B2	MII
mii_col	Bidir	E3	MII
mii_crs	Bidir	F3	MII
mii_mdc	Bidir	D3	MII
mii_mdio	Bidir	C3	MII
mii_rev_en	In	D2	MII Pull Up for reverse mode
mii_rmii_en	In	F2	MII Pull Up for reduced MII mode
mii_rx_clk	Bidir	B4	50MHZ common ref clock in RMII Mode
mii_rx_data[0]	Bidir	F4	MII
mii_rx_data[1]	Bidir	E4	MII
mii_rx_data[2]	Bidir	D4	MII
mii_rx_data[3]	Bidir	C4	MII
mii_rx_dv	Bidir	B3	MII
mii_speed_sel	In	E2	MII 10Mbs – Pull Down/100Mbs Pull Up (Reduced Mode only)
mii_tx_clk	Bidir	B5	MII
mii_tx_data[0]	Bidir	F5	MII
mii_tx_data[1]	Bidir	E5	MII
mii_tx_data[2]	Bidir	D5	MII
mii_tx_data[3]	Bidir	C5	MII
mii_tx_en	Bidir	A4	MII
uart1_sin	In	E1	UART 1
uart1_sout	Out	G2	UART 1
uart2_sin	In	F1	UART 2
uart2_sout	Out	G1	UART 2

Table 51: SDRAM pin descriptions

Signal	Type	No.	Function
mem_addr[10]	Out	AA22	SDRAM address
/mem_cas	Out	Y24	Column address strobe
/mem_cs	Out	AA24	Chip select
/mem_dqs[0]	Bidir	U26	Data strobes, differential
/mem_dqs[1]	Bidir	R26	Data strobes, differential
/mem_dqs[2]	Bidir	L26	Data strobes, differential
/mem_dqs[3]	Bidir	J25	Data strobes, differential
/mem_ras	Out	W24	Row address strobe
/mem_we	Out	AB26	Write enable
mem_addr[0]	Out	Y22	SDRAM address
mem_addr[1]	Out	Y23	SDRAM address
mem_addr[11]	Out	AC23	SDRAM address
mem_addr[12]	Out	AD25	SDRAM address
mem_addr[13]	Out	AD26	SDRAM address
mem_addr[2]	Out	AA23	SDRAM address
mem_addr[3]	Out	AB23	SDRAM address
mem_addr[4]	Out	AB24	SDRAM address
mem_addr[5]	Out	AC26	SDRAM address
mem_addr[6]	Out	AC25	SDRAM address
mem_addr[7]	Out	AC24	SDRAM address
mem_addr[8]	Out	AB22	SDRAM address
mem_addr[9]	Out	AC22	SDRAM address
mem_ba[0]	Out	W23	Memory Bank Address
mem_ba[1]	Out	W22	Memory Bank Address
mem_cke	Out	AB25	Clock Enable
/mem_clk0	Out	U22	Clock 0, differential
mem_clk0	Out	V22	Clock 0
/mem_clk1	Out	P26	Clock 1, differential
mem_clk1	Out	P25	Clock 1
mem_data[0]	Bidir	V24	SDRAM Data
mem_data[1]	Bidir	AA25	SDRAM Data
mem_data[10]	Bidir	T24	SDRAM Data
mem_data[11]	Bidir	T25	SDRAM Data
mem_data[12]	Bidir	T26	SDRAM Data
mem_data[13]	Bidir	R22	SDRAM Data
mem_data[14]	Bidir	R23	SDRAM Data
mem_data[15]	Bidir	R24	SDRAM Data
mem_data[16]	Bidir	P23	SDRAM Data
mem_data[17]	Bidir	N23	SDRAM Data
mem_data[18]	Bidir	N24	SDRAM Data
mem_data[19]	Bidir	N25	SDRAM Data
mem_data[2]	Bidir	Y25	SDRAM Data
mem_data[20]	Bidir	N26	SDRAM Data
mem_data[21]	Bidir	M23	SDRAM Data
mem_data[22]	Bidir	M22	SDRAM Data
mem_data[23]	Bidir	M24	SDRAM Data
mem_data[24]	Bidir	L23	SDRAM Data
mem_data[25]	Bidir	L22	SDRAM Data
mem_data[26]	Bidir	L24	SDRAM Data
mem_data[27]	Bidir	L25	SDRAM Data
mem_data[28]	Bidir	K23	SDRAM Data
mem_data[29]	Bidir	K26	SDRAM Data
mem_data[3]	Bidir	W25	SDRAM Data
mem_data[30]	Bidir	K25	SDRAM Data
mem_data[31]	Bidir	K24	SDRAM Data
mem_data[4]	Bidir	AA26	SDRAM Data
mem_data[5]	Bidir	Y26	SDRAM Data
mem_data[6]	Bidir	W26	SDRAM Data
mem_data[7]	Bidir	V25	SDRAM Data
mem_data[8]	Bidir	T23	SDRAM Data

Signal	Type	No.	Function
mem_data[9]	Bidir	T22	SDRAM Data
mem_dqm[0]	Out	U24	Data write mask
mem_dqm[1]	Out	P24	Data write mask
mem_dqm[2]	Out	M25	Data write mask
mem_dqm[3]	Out	H26	Data write mask
mem_dqs[0]	Bidir	U25	Data strobes
mem_dqs[1]	Bidir	R25	Data strobes
mem_dqs[2]	Bidir	M26	Data strobes
mem_dqs[3]	Bidir	H25	Data strobes
mem_odt	Out	V23	ODT Control
mem_shield	In	N22	See SDRAM 4.5 for special handling.
Mem_vref	In	P22	See SDRAM 4.5 for special handling.

Table 52 : Synchronization & JTAG pin descriptions

Signal	Type	No.	Function
sync_out	Out	A13	Synchronisation clock signal, output on master device. Connect to sync_in.
/halt	Open drain	B13	Driven low by any device that wishes to halt the system, could be asserted by an external device. Requires a pull-up resistor.
Run	In	A16	Driven by device master, indicates all devices should start running.
Arm_rtck	In	H3	Synchronization & JTAG
arm_tdi	In	D1	Synchronization & JTAG
arm_tdo	Out	G5	Synchronization & JTAG
arm_tms	In	C1	Synchronization & JTAG
arm_trstb	In	H5	Synchronization & JTAG
master	In	E14	Indicates the device is a synchronisation master. (Tie High)
step	In	A15	Driven by Synchronization device master, indicates all devices should Step for 1 cycle. (single Device treat as NC)
sync_in	In	A14	Synchronisation clock signal. Direct Input from sync_out
synchrorun	Bidir	B14	NC
tck	In	G26	Synchronization & JTAG
tmod	In	H23	Synchronization & JTAG
tms	In	H24	Synchronization & JTAG
xtal_in	In	B17	Synchronization & JTAG
/rst	In	J22	Sync Active Low RESET
xtal_in_uart	In	H2	Synchronization & JTAG
xtal_out	Out	B16	Synchronization & JTAG
tdi	In	J23	Test Data Input
/trst	In	J24	Async Active low RESET
xtal_out_uart	Out	J3	Synchronization & JTAG
arm_tck	In	J4	Synchronization & JTAG
tdo	Out	K22	Test Data Output
boot_mode[0]	In	G22	See Table 8: Handling of Boot Mode Pins
boot_mode[1]	In	H22	See Table 8: Handling of Boot Mode Pins

Table 53: Miscellaneous pin descriptions

Signal	Type	No.	Function
BALL MISSING	NA	A01	BALL MISSING
BALL MISSING	NA	A26	BALL MISSING
BALL MISSING	N/A	AF01	BALL MISSING
BALL MISSING	N/A	AF26	BALL MISSING
Reserved	NA	A02	NC
Reserved	NA	AA3	NC
Reserved	NA	AA4	Pull Down
Reserved	NA	AB01	NC
Reserved	NA	AB02	NC
Reserved	NA	AB02	NC
Reserved	NA	AB10	NC
Reserved	NA	AB11	Pull Down
Reserved	NA	AB18	NC
Reserved	NA	AB19	NC
Reserved	NA	AB21	Pull Down
Reserved	NA	AC1	NC
Reserved	NA	AC10	NC
Reserved	NA	AC11	Pull Down
Reserved	NA	AC18	Pull Down
Reserved	NA	AC19	Pull Down
Reserved	NA	AD02	Pull Down
Reserved	NA	AD10	Pull Down
Reserved	NA	AD11	Pull Down
Reserved	NA	AD18	NC
Reserved	NA	AD19	Pull Down
Reserved	NA	AD19	Pull Down
Reserved	NA	AE09	NC
Reserved	NA	AE10	NC
Reserved	NA	AE19	NC
Reserved	NA	AE20	NC
Reserved	NA	AF10	Pull Down
Reserved	NA	AF14	Pull Down
Reserved	NA	AF19	NC
Reserved	NA	AF20	NC
Reserved	NA	AF21	Pull Down
Reserved	NA	B13	Pull Up
Reserved	NA	D17	NC
Reserved	NA	D17	NC
Reserved	NA	F22	Hard Ground
Reserved	NA	G23	NC
Reserved	NA	Y04	Pull Down

6 Pin-Out & Package

6.1 Pin-Out

Table 54: 672 – ball PBGA pin assignment

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
A01	BALL MISSING	AA01	pdb0_dat[4]	AB01	Reserved	AC01	Reserved
A02	Reserved	AA02	pdb0_dat[0]	AB02	Reserved	AC02	pdb1_in0
A03	mii_rx_er	AA03	Reserved	AB03	pdb1_in1	AC03	pdb1_dat[15]
A04	mii_tx_en	AA04	Reserved	AB04	pdb1_dat[14]	AC04	pdb1_dat[10]
A05	ebi_data[5]	AA05	pdb1_dat[13]	AB05	pdb1_dat[6]	AC05	pdb1_dat[5]
A06	ebi_data[7]	AA06	vddadi	AB06	NC	AC06	NC
A07	ebi_data[10]	AA07	vddadi	AB07	NC	AC07	NC
A08	ebi_data[15]	AA08	vddadi	AB08	NC	AC08	NC
A09	gnd	AA09	vddadi	AB09	NC	AC09	NC
A10	ebi_data[24]	AA10	vddadi	AB10	Reserved	AC10	Reserved
A11	ebi_data[29]	AA11	vddadi	AB11	Reserved	AC11	Reserved
A12	/ebi_decode[2]	AA12	vddadi	AB12	NC	AC12	NC
A13	sync_out	AA13	vddadi	AB13	NC	AC13	NC
A14	sync_in	AA14	vddadi	AB14	NC	AC14	NC
A15	step	AA15	vddadi	AB15	NC	AC15	NC
A16	run	AA16	vddadi	AB16	NC	AC16	NC
A17	ebi_dreq[0]	AA17	vddadi	AB17	NC	AC17	NC
A18	gnd	AA18	vddadi	AB18	Reserved	AC18	Reserved
A19	ebi_dreq[1]	AA19	vddadi	AB19	Reserved	AC19	Reserved
A20	ebi_busy[2]	AA20	vddadi	AB20	NC	AC20	NC
A21	ebi_busy[0]	AA21	vddmem	AB21	Reserved	AC21	NC
A22	/ebi_oe	AA22	mem_addr[10]	AB22	mem_addr[8]	AC22	mem_addr[9]
A23	ebi_addr[1]	AA23	mem_addr[2]	AB23	mem_addr[3]	AC23	mem_addr[11]
A24	ebi_addr[2]	AA24	/mem_cs	AB24	mem_addr[4]	AC24	mem_addr[7]
A25	gnd	AA25	mem_data[1]	AB25	mem_cke	AC25	mem_addr[6]
A26	BALL MISSING	AA26	mem_data[4]	AB26	/mem_we	AC26	mem_addr[5]

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
AD01	pdb1_val	AE01	gnd	AF01	BALL MISSING	B01	gnd
AD02	Reserved	AE02	pdb1_dat[11]	AF02	gnd	B02	mii_addr[4]
AD03	pdb1_dat[12]	AE03	pdb1_dat[9]	AF03	pdb1_dat[3]	B03	mii_rx_dv
AD04	pdb1_dat[8]	AE04	pdb1_dat[7]	AF04	pdb1_dat[2]	B04	mii_rx_clk
AD05	pdb1_dat[4]	AE05	pdb1_dat[1]	AF05	pdb1_dat[0]	B05	mii_tx_clk
AD06	NC	AE06	NC	AF06	NC	B06	ebi_data[6]
AD07	NC	AE07	NC	AF07	NC	B07	ebi_data[9]
AD08	NC	AE08	NC	AF08	NC	B08	ebi_data[14]
AD09	NC	AE09	Reserved	AF09	gnd	B09	ebi_data[19]
AD10	Reserved	AE10	Reserved	AF10	Reserved	B10	ebi_data[23]
AD11	Reserved	AE11	NC	AF11	NC	B11	ebi_data[28]
AD12	NC	AE12	NC	AF12	NC	B12	/ebi_decode[1]
AD13	NC	AE13	NC	AF13	NC	B13	/halt
AD14	NC	AE14	NC	AF14	Reserved	B14	synchrorun
AD15	NC	AE15	NC	AF15	NC	B15	/ebi_decode[7]
AD16	NC	AE16	NC	AF16	NC	B16	xtal_out
AD17	NC	AE17	NC	AF17	NC	B17	xtal_in
AD18	Reserved	AE18	NC	AF18	gnd	B18	ebi_dreq[2]
AD19	Reserved	AE19	Reserved	AF19	Reserved	B19	ebi_busy[1]
AD20	NC	AE20	Reserved	AF20	Reserved	B20	ebi_wp[2]
AD21	NC	AE21	NC	AF21	Reserved	B21	/ebi_we
AD22	NC	AE22	NC	AF22	NC	B22	ebi_addr[0]
AD23	NC	AE23	NC	AF23	NC	B23	ebi_addr[4]
AD24	NC	AE24	NC	AF24	NC	B24	ebi_addr[3]
AD25	mem_addr[12]	AE25	NC	AF25	gnd	B25	ebi_addr[13]
AD26	mem_addr[13]	AE26	gnd	AF26	BALL MISSING	B26	gnd

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
C01	arm_tms	D01	arm_tdi	E01	uart1_sin	F01	uart2_sin
C02	mii_addr[3]	D02	mii_rev_en	E02	mii_speed_sel	F02	mii_rmii_en
C03	mii_mdio	D03	mii_mdc	E03	mii_col	F03	mii_crs
C04	mii_rx_data[3]	D04	mii_rx_data[2]	E04	mii_rx_data[1]	F04	mii_rx_data[0]
C05	mii_tx_data[3]	D05	mii_tx_data[2]	E05	mii_tx_data[1]	F05	mii_tx_data[0]
C06	ebi_data[3]	D06	ebi_data[2]	E06	ebi_data[1]	F06	vddio
C07	ebi_data[8]	D07	ebi_data[4]	E07	ebi_data[0]	F07	vddio
C08	ebi_data[13]	D08	ebi_data[12]	E08	ebi_data[11]	F08	vddio
C09	ebi_data[18]	D09	ebi_data[17]	E09	ebi_data[16]	F09	vddio
C10	ebi_data[22]	D10	ebi_data[21]	E10	ebi_data[20]	F10	vddio
C11	ebi_data[27]	D11	ebi_data[26]	E11	ebi_data[25]	F11	vddio
C12	/ebi_decode[0]	D12	ebi_data[31]	E12	ebi_data[30]	F12	vddio
C13	/ebi_decode[5]	D13	/ebi_decode[4]	E13	/ebi_decode[3]	F13	vddio
C14	Proc_clk	D14	/ebi_decode[6]	E14	master	F14	vddio
C15	ebi_ready	D15	ebi_clk	E15	pll_vdd	F15	vddio
C16	Proc_irq	D16	pll_vss	E16	pll_avdd	F16	vddio
C17	ebi_dreq[3]	D17	Reserved	E17	pll_avss	F17	vddio
C18	ebi_wp[0]	D18	ebi_dwidth[1]	E18	ebi_rp	F18	vddio
C19	ebi_wp[1]	D19	ebi_dwidth[2]	E19	ebi_dwidth[0]	F19	vddio
C20	ebi_byte_mode[1]	D20	ebi_byte_mode[2]	E20	ebi_byte_mode[3]	F20	vddio
C21	ebi_byte_mode[0]	D21	ebi_addr[6]	E21	ebi_addr[11]	F21	vddio
C22	ebi_addr[5]	D22	ebi_addr[9]	E22	ebi_addr[10]	F22	Reserved
C23	ebi_addr[7]	D23	ebi_addr[12]	E23	ebi_addr[16]	F23	ebi_addr[20]
C24	ebi_addr[8]	D24	ebi_addr[15]	E24	ebi_addr[19]	F24	ebi_addr[21]
C25	ebi_addr[14]	D25	ebi_addr[18]	E25	ebi_addr[22]	F25	ebi_addr[25]
C26	ebi_addr[17]	D26	ebi_addr[23]	E26	ebi_addr[26]	F26	ebi_addr[27]

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
G01	uart2_sout	H01	arm_gpio[7]	J01	gnd	K01	sd_gpio[31]
G02	uart1_sout	H02	xtal_in_uart	J02	arm_gpio[1]	K02	arm_gpio[0]
G03	mii_addr[0]	H03	arm_rtck	J03	xtal_out_uart	K03	arm_gpio[2]
G04	mii_addr[2]	H04	mii_addr[1]	J04	arm_tck	K04	arm_gpio[6]
G05	arm_tdo	H05	arm_trstb	J05	arm_gpio[4]	K05	arm_gpio[5]
G06	vddio	H06	vddio	J06	vddio	K06	vddio
G07	vddcore	H07	vddcore	J07	vddcore	K07	vddcore
G08	vddcore	H08	vddcore	J08	vddcore	K08	vddcore
G09	vddcore	H09	vddcore	J09	gnd	K09	gnd
G10	vddcore	H10	vddcore	J10	gnd	K10	gnd
G11	vddcore	H11	vddcore	J11	gnd	K11	gnd
G12	vddcore	H12	vddcore	J12	gnd	K12	gnd
G13	vddcore	H13	vddcore	J13	gnd	K13	gnd
G14	vddcore	H14	vddcore	J14	gnd	K14	gnd
G15	vddcore	H15	vddcore	J15	gnd	K15	gnd
G16	vddcore	H16	vddcore	J16	gnd	K16	gnd
G17	vddcore	H17	vddcore	J17	gnd	K17	gnd
G18	vddcore	H18	vddcore	J18	gnd	K18	gnd
G19	vddcore	H19	vddcore	J19	vddcore	K19	vddcore
G20	vddcore	H20	vddcore	J20	vddcore	K20	vddcore
G21	vddio	H21	vddio	J21	vddio	K21	vddmem
G22	boot_mode[0]	H22	boot_mode[1]	J22	/rst	K22	tdo
G23	Reserved	H23	tmod	J23	tdi	K23	mem_data[28]
G24	ebi_addr[24]	H24	tms	J24	/trst	K24	mem_data[31]
G25	ebi_addr[28]	H25	mem_dqs[3]	J25	/mem_dqs[3]	K25	mem_data[30]
G26	tck	H26	mem_dqm[3]	J26	gnd	K26	mem_data[29]

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
L01	sd_gpio[27]	M01	sd_gpio[22]	N01	sd_gpio[17]	P01	sd_gpio[12]
L02	sd_gpio[28]	M02	sd_gpio[23]	N02	sd_gpio[18]	P02	sd_gpio[13]
L03	sd_gpio[29]	M03	sd_gpio[24]	N03	sd_gpio[19]	P03	sd_gpio[14]
L04	sd_gpio[30]	M04	sd_gpio[25]	N04	sd_gpio[20]	P04	sd_gpio[15]
L05	arm_gpio[3]	M05	sd_gpio[26]	N05	sd_gpio[21]	P05	sd_gpio[16]
L06	vddio	M06	vddio	N06	vddio	P06	vddio
L07	vddcore	M07	vddcore	N07	vddcore	P07	vddcore
L08	vddcore	M08	vddcore	N08	vddcore	P08	vddcore
L09	gnd	M09	gnd	N09	gnd	P09	gnd
L10	gnd	M10	gnd	N10	gnd	P10	gnd
L11	gnd	M11	gnd	N11	gnd	P11	gnd
L12	gnd	M12	gnd	N12	gnd	P12	gnd
L13	gnd	M13	gnd	N13	gnd	P13	gnd
L14	gnd	M14	gnd	N14	gnd	P14	gnd
L15	gnd	M15	gnd	N15	gnd	P15	gnd
L16	gnd	M16	gnd	N16	gnd	P16	gnd
L17	gnd	M17	gnd	N17	gnd	P17	gnd
L18	gnd	M18	gnd	N18	gnd	P18	gnd
L19	vddcore	M19	vddcore	N19	vddcore	P19	vddcore
L20	vddcore	M20	vddcore	N20	vddcore	P20	vddcore
L21	vddmem	M21	vddmem	N21	vddmem	P21	vddmem
L22	mem_data[25]	M22	mem_data[22]	N22	mem_shield	P22	mem_vref
L23	mem_data[24]	M23	mem_data[21]	N23	mem_data[17]	P23	mem_data[16]
L24	mem_data[26]	M24	mem_data[23]	N24	mem_data[18]	P24	mem_dqm[1]
L25	mem_data[27]	M25	mem_dqm[2]	N25	mem_data[19]	P25	mem_clk1
L26	/mem_dqs[2]	M26	mem_dqs[2]	N26	mem_data[20]	P26	/mem_clk1

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
R01	sd_gpio[7]	T01	sd_gpio[2]	U01	NC	V01	gnd
R02	sd_gpio[8]	T02	sd_gpio[3]	U02	pdb0_dat[15]	V02	pdb0_dat[14]
R03	sd_gpio[9]	T03	sd_gpio[4]	U03	pdb0_dat[13]	V03	pdb0_dat[11]
R04	sd_gpio[10]	T04	sd_gpio[5]	U04	sd_gpio[0]	V04	pdb0_dat[8]
R05	sd_gpio[11]	T05	sd_gpio[6]	U05	sd_gpio[1]	V05	pdb0_dat[9]
R06	vddio	T06	vddadi	U06	vddadi	V06	vddadi
R07	vddcore	T07	vddcore	U07	vddcore	V07	vddcore
R08	vddcore	T08	vddcore	U08	vddcore	V08	vddcore
R09	gnd	T09	gnd	U09	gnd	V09	gnd
R10	gnd	T10	gnd	U10	gnd	V10	gnd
R11	gnd	T11	gnd	U11	gnd	V11	gnd
R12	gnd	T12	gnd	U12	gnd	V12	gnd
R13	gnd	T13	gnd	U13	gnd	V13	gnd
R14	gnd	T14	gnd	U14	gnd	V14	gnd
R15	gnd	T15	gnd	U15	gnd	V15	gnd
R16	gnd	T16	gnd	U16	gnd	V16	gnd
R17	gnd	T17	gnd	U17	gnd	V17	gnd
R18	gnd	T18	gnd	U18	gnd	V18	gnd
R19	vddcore	T19	vddcore	U19	vddcore	V19	vddcore
R20	vddcore	T20	vddcore	U20	vddcore	V20	vddcore
R21	vddmem	T21	vddmem	U21	vddmem	V21	vddmem
R22	mem_data[13]	T22	mem_data[9]	U22	/mem_clk0	V22	mem_clk0
R23	mem_data[14]	T23	mem_data[8]	U23	gnd	V23	mem_odt
R24	mem_data[15]	T24	mem_data[10]	U24	mem_dqm[0]	V24	mem_data[0]
R25	mem_dqs[1]	T25	mem_data[11]	U25	mem_dqs[0]	V25	mem_data[7]
R26	/mem_dqs[1]	T26	mem_data[12]	U26	/mem_dqs[0]	V26	gnd

Pin	Signal	Pin	Signal
W01	pdb0_dat[12]	Y01	pdb0_dat[7]
W02	pdb0_dat[10]	Y02	pdb0_dat[5]
W03	pdb0_dat[6]	Y03	pdb0_dat[1]
W04	pdb0_dat[2]	Y04	Reserved
W05	pdb0_dat[3]	Y05	pdb0_val
W06	vddadi	Y06	vddadi
W07	vddcore	Y07	vddcore
W08	vddcore	Y08	vddcore
W09	vddcore	Y09	vddcore
W10	vddcore	Y10	vddcore
W11	vddcore	Y11	vddcore
W12	vddcore	Y12	vddcore
W13	vddcore	Y13	vddcore
W14	vddcore	Y14	vddcore
W15	vddcore	Y15	vddcore
W16	vddcore	Y16	vddcore
W17	vddcore	Y17	vddcore
W18	vddcore	Y18	vddcore
W19	vddcore	Y19	vddcore
W20	vddcore	Y20	vddcore
W21	vddmem	Y21	vddmem
W22	mem_ba[1]	Y22	mem_addr[0]
W23	mem_ba[0]	Y23	mem_addr[1]
W24	/mem_ras	Y24	/mem_cas
W25	mem_data[3]	Y25	mem_data[2]
W26	mem_data[6]	Y26	mem_data[5]

6.2 Package Diagram

Figure 52: PC202 pin orientation and footprint

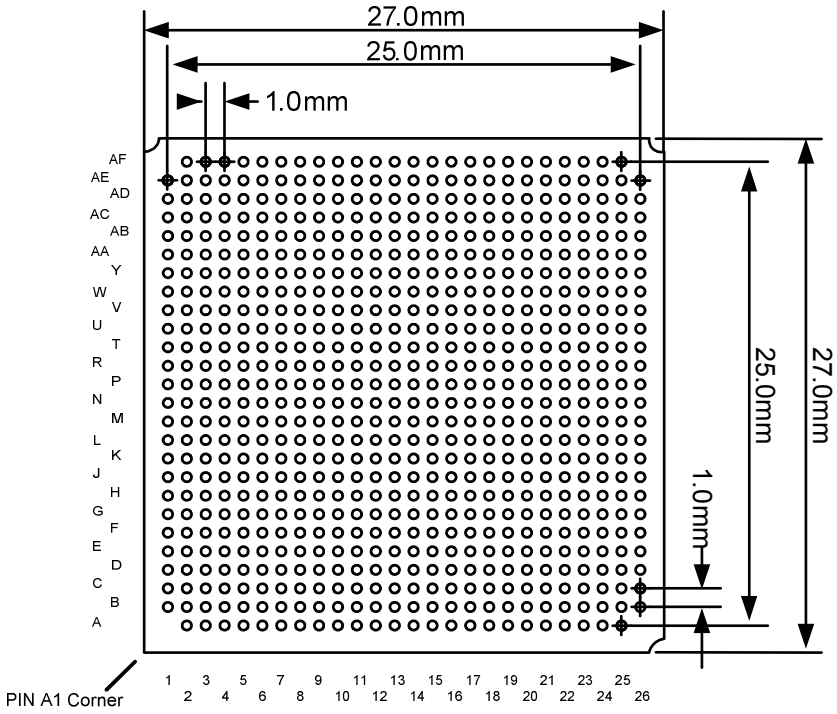


Figure 53: PC202 package mechanical drawing – Top View

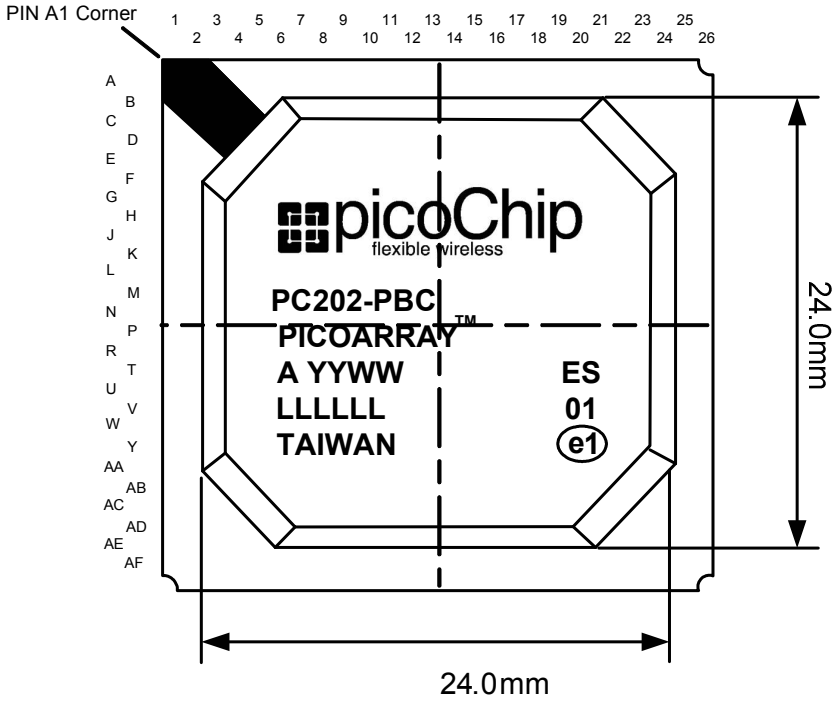


Figure 54: PC202 pin orientation and footprint – Profile

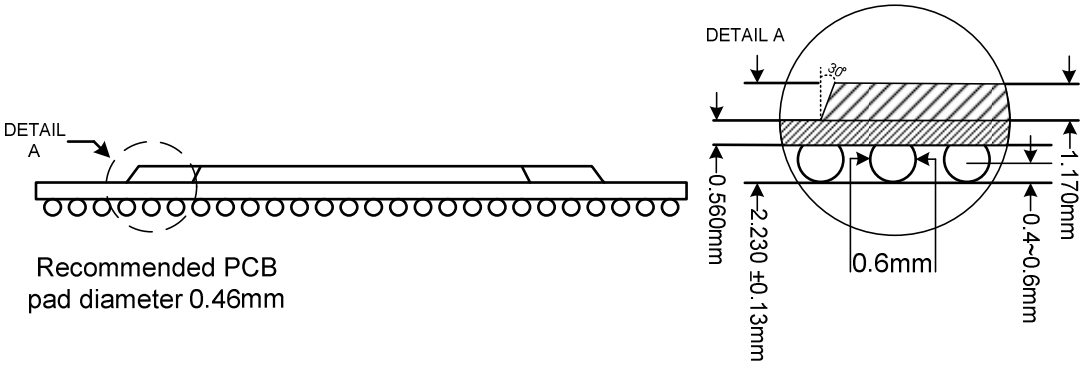


Figure 55: Pin orientation – Top View.

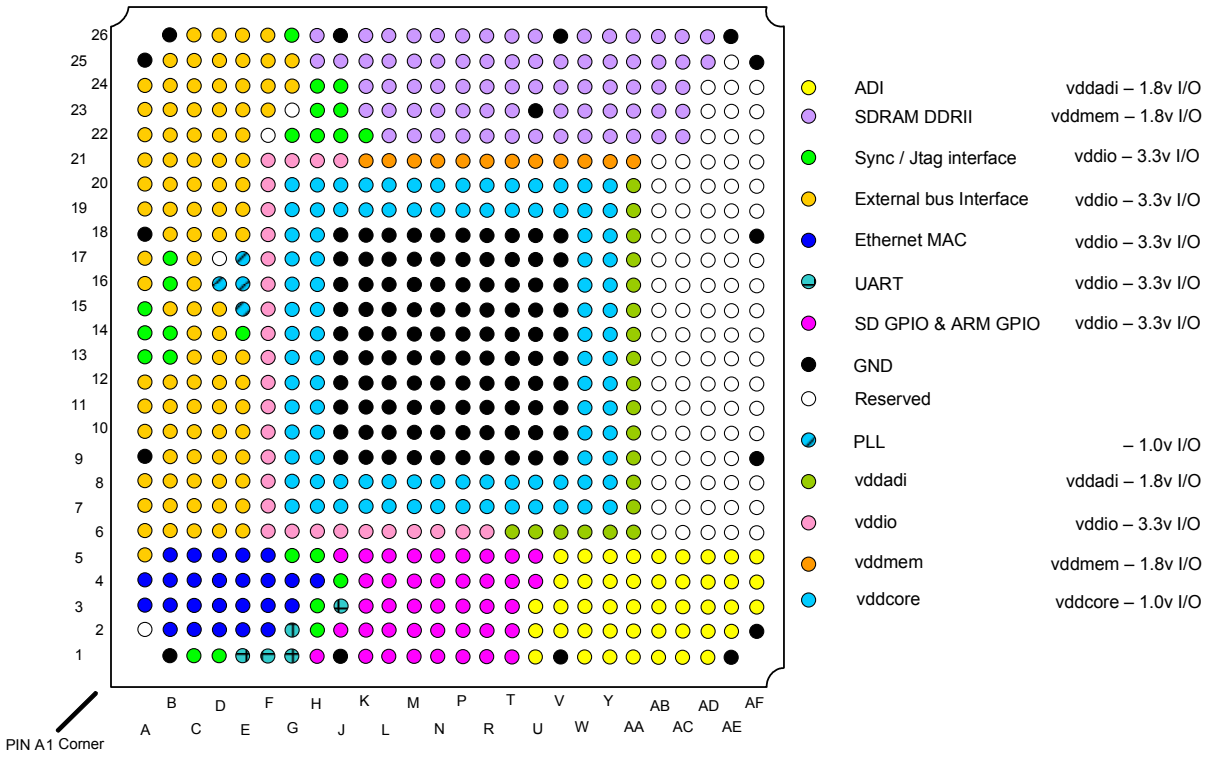
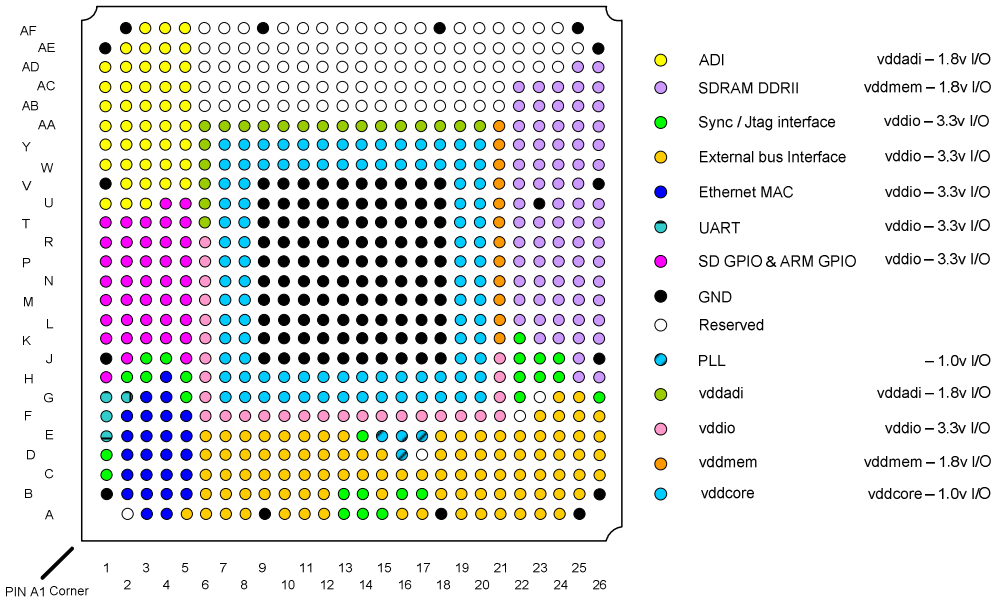


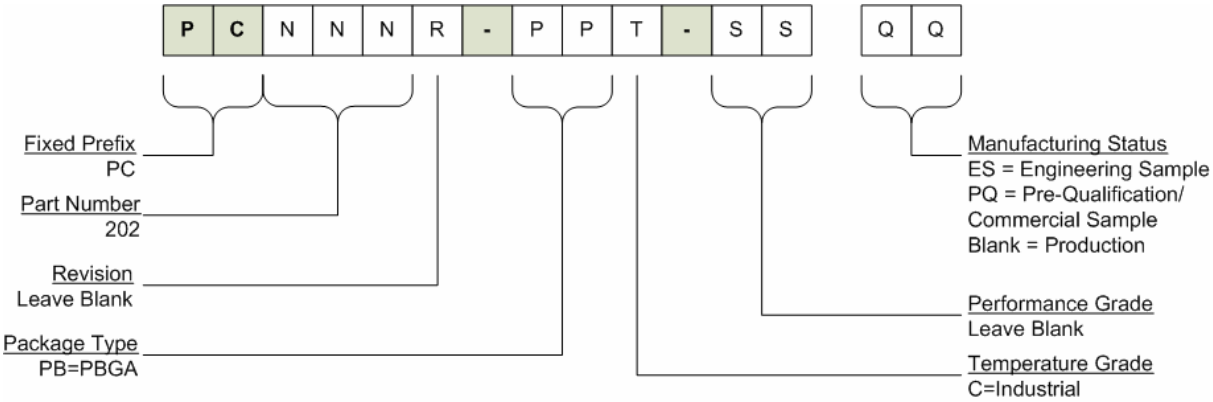
Figure 56:Pin orientation – Ball View



6.3 Reflow profile.

The PC202 reflow profile is in accordance with JEDEC JESD-020C for lead-free devices. The tpeak temperature for reflow is 250degC+0C/-5C as per the standard. The PC20x package is also MSL3 compliant and has been qualified using three reflow cycles with this tpeak value.

7 PC202 Part Number Description



Ordering Number	Package	Temperature Grade
PC202-PBC	672 PBGA	Industrial Temperature (0° to 85°C)

CAUTION ESD (electrostatic discharge) sensitive device.

Electrostatic charges on operators and equipment can discharge without detection, and may cause permanent damage to the device.

ESD precautions are recommended to avoid performance degradation or loss of functionality.

8 Glossary, Acronyms and terminology

@rate	"At rate" i.e., slot rate for inter AE communications
ADC	Analogue to digital converter
ADI	Asynchronous Data Interface
AE	Array Element
Async.	Asynchronous
Bidir.	Bidirectional
BGA	Ball Grid Array
Bus slot	Time divisions on the picoBus™
Config.	Configuration
CLK	Clock
CTRL	An AE optimized for control applications
CS	Control and synchronization
DAC	Digital to analogue converter
DMA	Direct Memory Access
DREQ	DMA Request
FAU	Function Accelerator Unit
F_{CLK}	The input clock, (not the microprocessor interface external clock)
FEC	Forward Error Correction
FHT	Fast Hadamard Transform
FIFO	First in, first out (memory)
FPGA	Field Programmable Gate Array
F_{SYS}.	The internal clock related to F_{CLK}
FDD	Frequency domain duplex
FR	Full rate
GND or gnd	Ground
GPR	General purpose register
HLT	Halt
HR	Half rate
I/O	Input and output
IPC	Inter processor communications
IPI	Inter-picoArray™ Interface
IRQ	Interrupt Request
ITS	Interrupt Status (register)
JTAG	Joint Test Action Group that developed the IEEE 1149.1 standard
L1	Layer 1 or physical layer
LIW	Long Instruction Word
LSB	Least significant bit
MAC	Multiply Accumulate
MEM	An AE optimized for memory intensive applications
M/S	Master/Slave
MSB	Most significant bit
OE	Output Enable
picoArray™	The Picochip Designs Ltd. Proprietary array processing architecture
picoBus™	The bus connecting array elements on a picoArray. It is time divided into bus slots. The connectivity of signals is made by the switch Aes within these slots
PLL	Phase locked loop

Preliminary change	Device design information provided prior to device availability and subject to
PROM	Programmable Read-only Memory
PWR	Power
RISC	Reduced Instruction Set
RST	Reset
SDRAM	Synchronous Dynamic Random Access Memory
SERDES	Serializer Deserializer
Slot Rate	The regularity at which bus slots occur, which by implication specifies the maximum frequency of the signal it is assigned to. Slot rates must all be a power of 2 in the range 1-1024. e.g. a slot rate of eight means that the signal has a bus slot every eight cycles. Slot rates are declared on a signal with the notation @n where n is the slot rate
SRAM	Static Random Access Memory
STAN	Standard AE type
Sync.	Synchronization
uP or UP	Microprocessor
WE	Write Enable

9 Version History

Version	Date	Author(s)	Changes made
1.0	11/16/2006	RR	Initial Release
1.1	29/12/2006	DM	Miscellaneous edits, add jitter spec, osc thresholds
1.2	10/2/2007	RR	Updated EBI pin Descriptions
1.3	15/2/2007	RR	Updated pin descriptions
1.4	20/2/2007	RR	Updated Reduced MII information relating to speed select and MII timing parameters.
1.5	02/03/2007	RR	Updated first three pages
1.6	13/03/2007	RR	Add reflow profile information
1.7	20/04/2007	RR	Correct ebi_dwidth naming and ebi address aliasing
1.8	23/04/2007	RR	Add Proc_IF chapter to the Datasheet & Remove ADI Tandem mode
1.9	30/05/07	RR	Update MII diagrams
2.0	04/07/07	RR	General update – proc_if , ADI, power figures
2.**1**	03/10/07	RR	Increase Junction Temperature 85 to 100 Deg C