A Survey of Microprocessor Communication Methods

Kai S.Ng, student, IEEE

Department of Electrical and Electronic Engineering
Newcastle Polytechnic, Ellison Place
Newcastle upon Tyne, NE1 8ST, United Kingdom

Abstract

This paper describes a wide variety of communication methods, both bit serial and bit parallel, used in current microprocessor and microcontroller designs. Nine different coprocessor interfaces are illustrated. The practical examples of burst and pipeline access (as featured in Am29000), data synchronization, bus snooping (MC68040), address recognition and resolution of masters covered in this article provide a basis for further studies of interface architectures for inter chip communication.

Introduction

The bus interface partitions of microprocessors are increasingly sophisticated as new architectures are being introduced to increase the throughput between the CPU chip and the rest of the system. A bit parallel architecture is generally employed for interface with the main semiconductor RAM. A bit serial approach is adopted if the intended use of the interface requires an economical solution for exchange of digital control messages in a multi processor/controller environment.

The typical bit parallel interface of a microprocessor consists of a data bus, an address bus and a set of control lines. On the circuit board, the control lines provide connections for features like interrupts, DMA and communication with coprocessors. This paper illustrates the operations of 8 different coprocessor interfaces.

The incorporation of a bit serial port in a microprocessor chip is cost effective because the serial port adds only a handful of pins to a microprocessor chip, which typically takes up between 40 to 168 pins (e.g. the i486), as opposed to providing another chip on the circuit board. Preferably if the serial port is implemented as an integral feature within the CPU chip boundary it should be optimised with other hardware design features such as the instruction set and exception logic to extract the highest performance from the port. The transputer demonstrates that the bit-serial method offers a simple and very cost effective high performance solution for inter processor communication. Another serial interface (I2C from Philips Components) considered in this survey illustrates a simple 2-wire interface for connecting the microprocessor system master to other electronic application devices.

Coprocessor interface methods used in Intel 8*86 systems

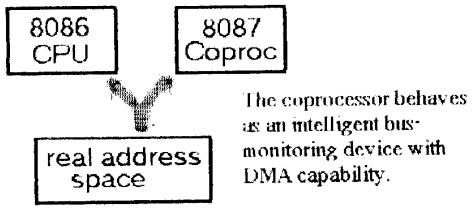

Fig.1. A coprocessor with bus monitoring and DMA.

In Intel's dated 8086 architecture, the 8087 floating point coprocessor continuously monitors the information stream on the system bus and becomes active if a coprocessor instruction is decoded. When the floating point operation is completed, the result is written back into memory by DMA. The problem of data synchronization due to independent memory accesses by both CPU and coprocessor is resolved at assembler or compiler level by inserting the appropriate number of wait instructions [1].

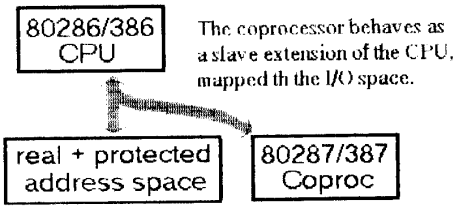

Fig.2. The slave coprocessor is supervised.

0-8186-2760-3/92 $03.00 © 1992 IEEE

In the 80286/386 architecture which has a powerful range of memory management and protection features, bus monitoring ceases to be practical [2],[3]. All 80287 coprocessor memory transactions are performed automatically through the CPU which has access to the register stack of the coprocessor. These registers are mapped into a reserved portion of the I/O space.

The 82786 integrated graphics and display coprocessor employs message passing for interface with the 80286/386 processor [4]. Here the CPU writes a sequence of drawing commands to the system memory and flags the coprocessor to begin. The coprocessor then fetches the commands from the system memory by DMA and interact with its own graphics display memory. A glue circuit is required if processor access to both types of memory is required.

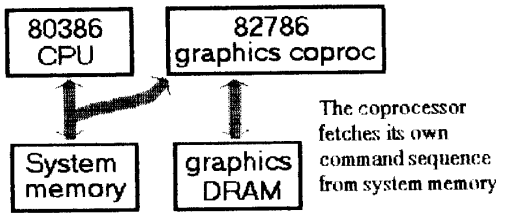

Fig.3. An message passing DMA coprocessor.

The interface of the 82380 DMA and interrupt controller does not require any glue logic [5]. This device has no chip select pin; its internal registers span 256 consecutive locations in the I/O address space, the base offset of which is controlled by software. In slave mode the 82380 monitors the address bus for addresses within this range and acts on any bus cycle that accesses an internal register. Upon reset the register set is mapped from address zero. By writing to the relocation register at port address 7F, the physical location of all registers including this one will be moved [6].

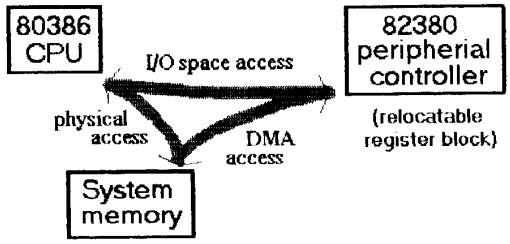

Fig.4. A dynamically relocatable intelligent device.

Bus interface features of Motorola MC680*0

The Motorola MC68020/30 architecture treats the coprocessor as a peripheral chip with a set of registers in a separate address space, distinguished from normal memory space by the CPU signalling a status code "111" on the status lines [1],[2]. This set of 32 byte locations are called "Bus-interface registers", or BIR, via which the CPU communicates with the functional registers of the coprocessor. A multiplicity of eight memory page blocks are reserved for interface with up to eight coprocessors, each of which have identical BIR. The coprocessor talks to the CPU not by hardware but via one of the BIR, called the response register, which the CPU should refer to check if the coprocessor has completed the operation assigned. No special hardware handshake control lines are required for this method of interface (unless the coprocessor has DMA facility).

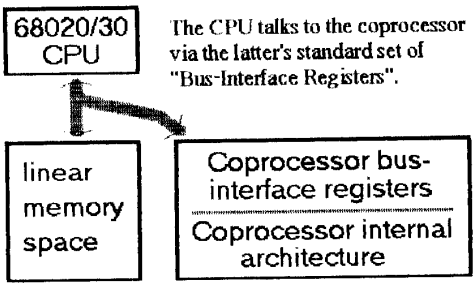

Fig.5. The use of Bus-interface registers permits a standard hardware protocol between CPU and coprocessors.

The latest microprocessor in the family, the MC68040, has numerous performance enhancing capabilities. A particular innovative feature is the support for more than one bus master via a technique known as bus snooping which synchronises the data held in RAM to those captured in the cache inside the CPU [7]. Here when there is a DMA memory write access, the CPU monitors the address line to see whether a cache line is matched, and if so the cache is either updated or marked invalid. If the DMA access is a read, the CPU will intervene to supply the data from the cache.

Interfacing coprocessors to National Semiconductor's NS32000

Coprocessors for National Semiconductor's NS32032 do not execute concurrently [1],[2],[8]. When the CPU detects a coprocessor instruction, "1101" is asserted on the status lines to pass over the command and any operand the coprocessor requires. The coprocessor then begins execution, during which time the CPU performs instruction prefetch operations until its buffer is filled Then "0011" is placed on the status lines to indicate it will wait for the coprocessor to finish. When the coprocessor is ready it

pulses the pin function AT/SPC and the CPU responds by reading the status word of the coprocessor. Finally the finished result is loaded into the CPU.

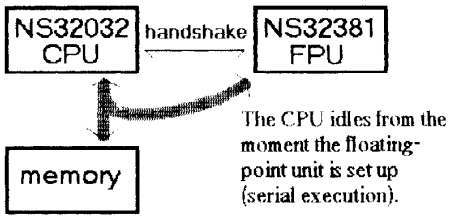

Fig.6. The NS32381 processor extension executes serially.

The protocol described above, as used for the NS32381 floating point coprocessor, is inefficient because of the amount of CPU-FPU communication overhead. This problem has been addressed by a new CPU member of the microprocessor family, the NS32532, which teams up with NS32580 controller to interact with Weitek's pipelined WTL3164 floating point chip [9]. The Weitek chip has three floating point data paths which permit up to three commands to overlap in their execution. The function of the NS32580 is to maintain the supply of floating point instructions at the high rate the WTL3164 can accept via its FIFO.

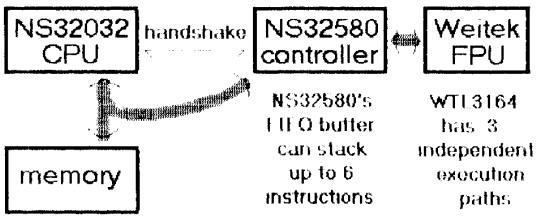

Fig.7. Piplelined coprocessor execution reduces CPU-FPU interface overhead.

Interfacing coprocessors to Zilog Z80000

The Zilog Z80000 processor can detect up to four external coprocessors [1],[2]. A coprocessor monitors the bus until it recognises a coprocessor instruction pattern for itself and is permitted to act by the CPU asserting the status lines (e.g. a status code of "1010" means the bus cycle type is a cacheable data transfer between coprocessor and memory). Unlike the 8086 the architecture does not allow a coprocessor to have DMA, so data must be passed via the CPU addressing mechanism. Thus the CPU has total control over the data flow to and from the coprocessors. The use of bus monitoring for fetching instructions and leaving the results until the CPU wants it is appropriate in the Zilog architecture because all memory management features are provided by the CPU chip.

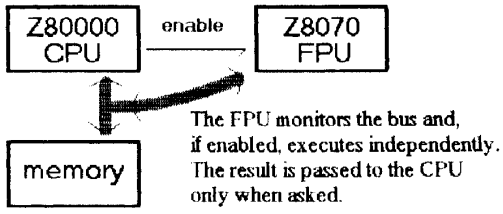

Fig.8. The Zilog coprocessor monitors the bus but does not take over it.

Streamline bus interface of Advanced Micro Device's Am29000

When it was first released Advanced Micro Device's Am29000 microprocessor demonstrated some innovative architectural features to sustain very high bus throughput; its multi-cycle burst-mode memory accesses can stream concurrently along separate 32 bit data and instruction buses [10],[11]. The Am29000 attempts to use burst-mode accesses whenever it performs instruction prefetches or load/store multiple operations. When the processor desires a burst mode access it asserts the pin function /IBREQ or /DBREQ during the cycle in which the initial address is placed on the bus. A system supporting burst-mode would acknowledge the processor by asserting the appropriate pin function /IBACK or /DBACK. If the first access does not establish a burst mode access, the processor continues the attempt on each subsequent address transfer.

The utility of the non multiplexed address bus is maximised in the Am29000 architecture by overlapping bus accesses which require more than one machine cycle. A pipeline access is one that starts bus activity while an earlier access, called primary access, is still in progress. The pin function /PEN is driven by the address latch of the memory system when pipeline is supported, which, if asserted prompts the processor to pipeline the next access while the primary access is still taking place. When the CPU completes the primary accesss the pipelined access becomes the next primary access. The pipeline access on the bus is cancelled immediately if the processor takes an interrupt or trap before the access becomes primary.

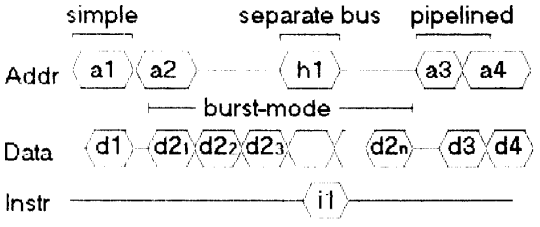

Fig.9. Am29000's four modes of bus access.

Having been the major manufacturer of parts for bit slice computers, A.M.D. is well placed to design the burst mode and pipelined bus access features in the first microprocessor of their own design. These features have since become commonplace on new generations of competitive high performance microprocessors released thereafter [12].

Interfacing coprocessor to VLSI Technology's VL86C020

VLSI Technology's most recent microprocessor the VL86C020, has 4K fixed instruction and data cache on-chip and provides a separate 32-bit coprocessor interface bus in addition to non-multiplexed address and data buses [13],[14]. For concurrent operations to be carried out most effeciently, the coprocessor requires direct access to the cache for instructions and data, and this is achieved via the separate, exclusive, coprocessor bus. Up to 16 coprocessors, some of which are suggested to be custom designs, are addresses by a 4-bit field within the 32-bit RISC type instruction code.

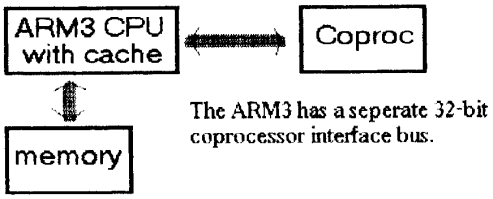

Fig.10. ARM3's separate coprocessor bus solution.

The interface of transputer links

The INMOS transputer microprocessor device [15], such as IMST805, has a standard local memory bus and four identical high speed (up to 20 Mbits/sec) serial links. Transputer serial links are used for node-to-node interconnect of several microprocessors. Each transputer can be scheduled to execute several processes, switching over from one to another when waiting for data from a link. The instruction set and the firmware of the transputer are purposely designed to support concurrency at hardware level. Thus the descheduling of a process is done very fast; typically requiring only 19 processor cycles.

The link consist of two uni-directional wires, the protocol over which multiplexes data and control information (there is no need for device address information). Messages are transmitted as a sequence of bytes, each of which must be acknowledged before the next transmission. A message byte is preceded by two start bits marked "11" and terminated by a "0" stop bit. An acknowledgement indicates that the data has been used by the receiving process and that the (single byte) buffer is ready to receive another byte. If the receiving process is waiting to take data immediately it is passed over the link, the protocol permits an acknowledgement as soon as the start bits of a data packet is identified. Thus it is possible to increase the data considerably for later transputers which implement this overlap features.

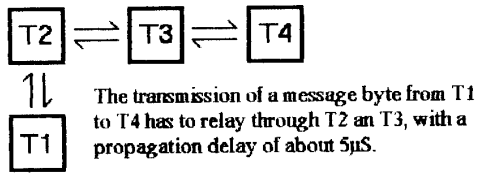

Fig.11. The data path in a transputer network is predetermined out by the compiler.

In a fixed array of T805s, the channel routing of data communication over the transputer links is managed by the compiler. On the most release of the transputer, the H1, the transmission of a message packet also includes a header which is used to identify the virtual channel of the destination transputer. A new network routing device, the C104, can handle 32 links and perform the routing of H1 messages automatically by hardware [16]. A number of C104s can be connected together to form a large dynamic routing network.

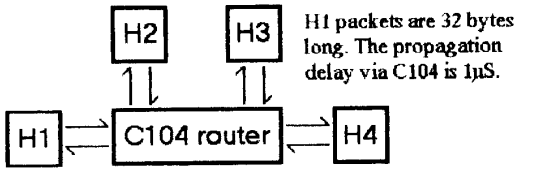

Fig.12. The new H1 transputers works with a router directly.

Philips Components' Inter-IC bus

To maximise the cost efficiency of hardware when designing digital control circuits for industrial, consumer and telecommunication equipments, Philips Components developed a bidirectional 2-wire bus called the I2C (Inter-IC) bus [17], for direct connection between microcontrollers and application devices like I/O ports, LCD drivers, serial RAM, audio communication and video tuners. A transmission rate of 100kHz between devices of different technologies is specified. The two pin functions are

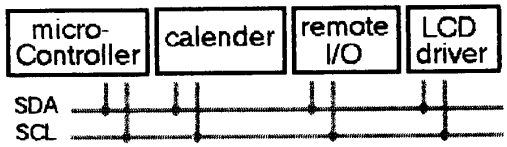

Fig.13. The simple architecture of Philips Inter-IC bus.

designated SDA for serial data and SCL for serial clock. These lines are required to be pulled up. During transmission, the master generates the clock pulses on SCL. For a data bit to be valid, its value on SDA must be stable during the high period of a clock pulse.

A transmission protocol between the bus master and the slave device begins with the "start" condition, defined as a negative edge on SDA when SCL is high. A seven bit address (most significant bit first) is then emitted, followed by a R/W bit. The slave device being addressed should then send a low bit for acknowledgement. Data is then transferred on a byte by byte basis, each requiring an acknowledgement bit asserted low from the receiving device, unless the data byte is the last one to be transmitted, in which case the acknowledgement is not generated. Whenever the acknowledgement is not asserted after the transfer of a data byte, the transmitting device must release SDA and the master generates the "stop" condition, defined as a positive edge when SCL is high.

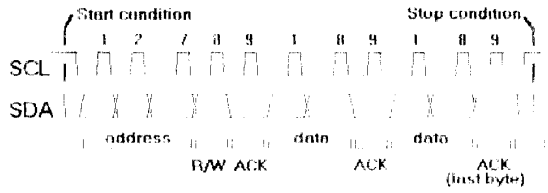

Fig.13. I2C bus protocol.

To enable slower devices to cope with fast data transfers, slaves can temporarily hold the SCL line low to force the master into a wait state. If more than one master uses the bus at the same time, communication between all devices continue normally until a master loses arbitration by identifying data inconsistency at the output. A master loses and switches to slave mode when it detects its own transmission of a logic high is being forced low by another competing master.

A wide range of peripheral ICs are manufactured, each type belonging to a predefined address group. Where several identical ICs are likely to be used together, programmable pin functions are used to provide hardware addressing bits. During reset, a type of addressing, the "general call" (distinguished by having a format of all "0"s for the first byte of the transmission), is used by the configuring master to prepare the system. Another type of addressing, the "hardware general call", is used when a hardware master IC, such as a keyboard scanner, does not have a slave to address to. Instead it generates its own address (in the second byte after emitting the header code for general call) which can be recognised by the intelligent microcontroller IC connected to the bus. In some systems, a dump address can be assigned by software into the hardware master IC to enable it to use the slave addressing protocol when it desires to use the bus to transmit new information.

National's HPC16400 microcontroller

Called a High Performance Communications microcontroller, National Semiconductor's HPC16400 has 4 serial ports: a proprietary Microwire/Plus interface, a full duplex UART with wake up feature, and two HDLC (High level Data Link Control) channels optimised for ISDN applications [18].

The Microwire interface consists of three pin functions SI, SO, SK, standing for serial input, output and clock. Communication is achieved via a single 8 bit shift register where data on SI loads to the least significant bit, and SO takes data from the most significant bit. The SK clock is either provided externally when the processor is in slave mode, or generated internally (up to 1MHz rate with a 16MHz processor) when it is in master mode. Software is used to program a control bit register to determine whether the processor is the master or slave. Serial data at SI is latched on the rising edge of SK. Data is transmitted at SO on the negative edge of SK.

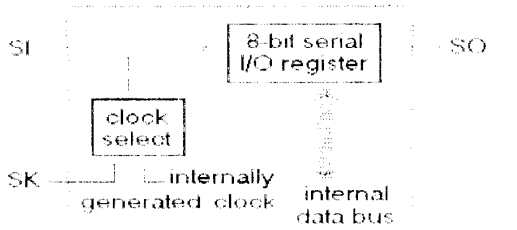

Fig.15. Block diagram of Microwire interface.

National Semiconductor supplies a number of peripheral devices with Microwire interfaces. When many of these are used in a system, the method for addressing an individual chip is to operate a microcontroller output port pin as a select line hardwired to the chip enable pin of the peripheral, and program the port when selecting the chip.

The conventional UART port of HPC16400 has five registers and five pin functions (receive data, transmit data, clock, external clock, and external clock select) to enable it to perform full duplex serial communication of up to 208.3Kbps. By operating the UART in the 9 bit wake up mode, it is possible to connect many processors in a network. In such application, a message consists of an address and data. A "1" on the ninth bit distinguishes an address frame from a data frame. An interrupt is signalled to the processor core when an address frame is detected. The processor then examines the receive buffer to decide whether to accept subsequent data frames.

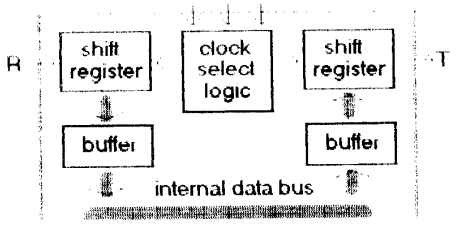

Fig.16. Block diagram of UART interface.

To support a wide range of communication applications, two 4.65Mbps HDLC ports are provided. Each HDLC port controls of two DMA channels and has five associated pin functions: channel clock input HCK; receive data, with data accepted on negative edge of HCK; transmitter output, with data clocked on positive clock edge; receiver enable; and transmitter enable.

During transmission, data is transferred byte by byte from external memory through the transmit buffer register where it is passed to an 8-bit shift register. The hardware computed CRC and the required control flag or idle pattern information selected by the transmit command are included automatically in the transmission stream before the frame is closed. It is possible to program the length of a message packet down to single bits.

In the receiver section of a HDLC port, data is shifted through two 8-bit serial shift registers before being loaded into the buffer register. The first two bytes after the opening flag of a frame are address information, and if there is a match with the two user programmable frame-address recognition bytes, the rest of the transmission goes to the DMA channel.

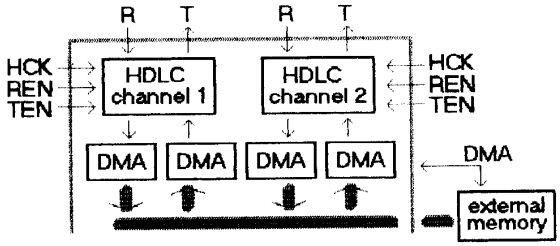

Fig.17. Block diagram of HDLC interface.

Summary

Many different methods for microprocessor communication have been presented. Where a microprocessor is selected for design into a consumer product, a bit-serial interface like I2C or Microwire is ideal for controlling electronic devices at minimal cost. In high-speed parallel bus environments, it is interesting to note how the technique of bus watching as used on the 8087 coprocessor became inapplicable in the 80287/387 architectures designed to superseded it, but is now being used again on the MC68040 which has embedded data cache. While at the top end of processor performance spectrum the trend in the use of burst mode and pipeline features is likely to influence the interface design of future RAM produces, at the other end, the bit-serial EEPROM [19], because of its low cost compact package, is being used more and more in embedded applications.

References

[1] G.D. Beals, "Extending microprocessor architectures," Byte, May 1985, pp.185 198.

[2] L. Ciminiera, and A. Valenzano, "Advanced microprocessor architecture," Addison-Wesley 1987.

[3] D. Perimutter, and A.K.W. Yuen, "The 80387 and its applications," IEEE Micro, Aug. 1987 pp.42-57.

[4] J. Nelson, "Simplifying graphics design with a highly integrated coprocessor," New Electronics, 10 June 1986, pp.32-38.

[5] I. Wilson, "Extending 80386 performance," New Electronics, 31 Mar. 1987, pp.30-33.

[6] "Microprocessors vol.2," Intel, Jan. 91.

[7] "MC68040 user's manual," Prentice Hall, 1990.

[8] "Microprocessor series 32000 and NSC800 family databook," National Semi. Corp., Aug. 1989.

[9] S. Iacobovici, "A pipelined interface for high floating point performance with precise exception," IEEE Micro, June 1988, pp.77-87.

[10] M.Johnson, "System considerations in the design of the Am29000," IEEE Micro, Aug. 1987, pp.28-41.

[11] J.D. Everitt, "Inexpensive performance using the Am29000," Microprocessors and Microsystems, July 1990, pp.407-415.

[12] H. Mitchell, "32-bit microprocessors," Blackwelll, 1991.

[13] S.B. Furber, and A.R.P. Thomas, "ARM3 a study in design for compatibility," Microprocessors and Microsystems, July 1990, pp.407-415.

[14] "Acorn RISC machine family data manual," Prentice Hall, 1990.

[15] "Transputer databook," INMOS Limited, 1989.

[16] C. Dyson, "INMOS H1 architecture revealed," New Electronics on Campus, special supplement to New Electronics, Spring 1991

[17] "I2C bus compatible ICs, data book 4 part 12a, "Philips Components technical handbook, 1989.

[18] "Microcontroller databook," National Semiconductor Corp., July 1989.

[19] R.A. Quinnell, "Serial memory offers cheap frills," EDN 1 Oct. 1991, pp.57-64.