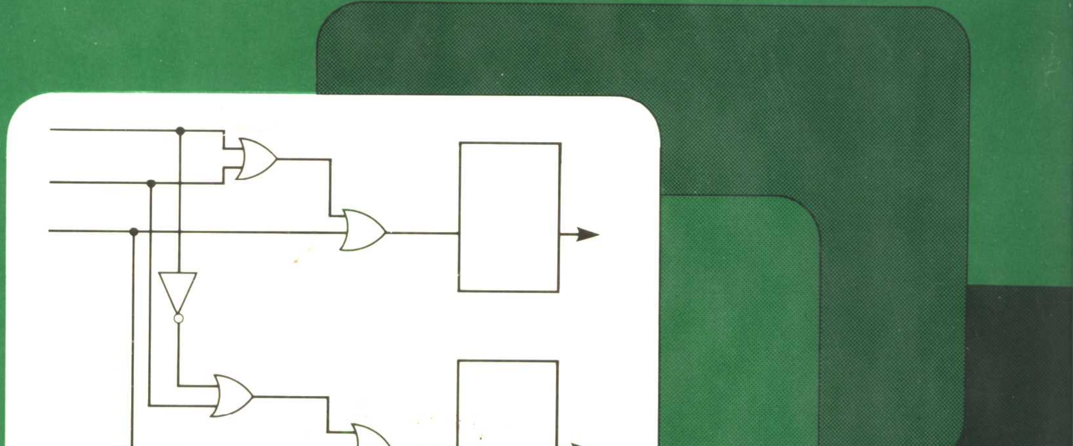
Practical Microprocessor Interfacing



S.A.Money

Practical Microprocessor Interfacing

S. A. Money
TEng (CEI) MIElecIE MBCS

Collins
Professional
Books

8 Grafton Street London W1X 3LA

Collins Professional Books
William Collins Sons & Co. Ltd
8 Grafton Street, London W1X 3LA

First published in Great Britain by
Collins Professional Books 1987

British Library Cataloguing in Publication Data
Money, Steve A.
Practical microprocessor interfacing.
1. Computer interfaces 2. Microprocessors
I. Title
004.6'16 TK7887.5

ISBN 0-00-383329-1

Typeset by Katerprint Typesetting Services, Oxford
Printed and bound in Great Britain by
Mackays of Chatham, Kent

Preface

Modern electronic equipment is frequently based around the use of a microprocessor which provides the major part of the control logic. In order to be useful any microcomputer based system must provide some means of getting signals into and out of the microcomputer, to allow communications with the outside world. The more popular general purpose microprocessor chips such as the Z80, 8085 and 6800 series provide only a basic Central Processing Unit (CPU) with no direct facilities for input and output of signals. All of these microprocessors operate with a set of bus systems which provide signal links between the CPU, memory and any other devices attached to the system. Some form of interface circuit is normally required to allow signals on the microcomputer bus system to be coupled to external equipment. In this book we shall be looking at the techniques for interfacing between the microprocessor bus system and the outside world.

The first two chapters of the book take a brief look at the structure and timing of the signals on the microprocessor bus system. All microprocessors need some form of memory system to hold the program instructions and any data to be processed. In Chapter 2 the basic principles of memory devices are examined and their connection to the bus system for the various types of processor is explained.

Perhaps the most straightforward method of getting signals into and out of a microprocessor system is by using a parallel interface. In Chapter 3 the basic principles of simple parallel input and output ports are discussed, and the techniques of interfacing these ports to the CPU bus system are examined. Some processors, such as the Z80 and 8085, have dedicated instructions and a separate addressing system for handling input and output data transfers, whilst the 6500 and 6800 series processors treat all input and output devices as if they were locations in the memory system. The arrangements for handling both dedicated and memory mapped I/O systems are examined. Later the chapter goes on to look at programmable input–output port devices, and some of the specially designed parallel interface chips.

Some typical applications of parallel input–output ports, including the connection of switches and keyboards and the drive circuits of lamps and relays, are examined in Chapter 4. The principles of driving stepper type motors are also explained, and the chapter goes on to look at two standard parallel interface schemes which are commonly used with microcomputer systems. The first is the Centronics parallel interface, which is used to provide a parallel output channel for driving printers and plotters. The second standard interface dealt with is IEEE488 General Purpose Interface Bus. This is a parallel data bus system, used for connecting laboratory instruments and similar equipment to a microcomputer system. The operation and organisation of the IEEE488 bus is explained, and a simple method of implementing this type of bus is given.

When signals are to be transmitted over long distances, or via a radio link, the serial type of data transmission format is generally used. In Chapter 5 the basic principles of serial transmission are explained. Both asynchronous and synchronous forms of transmission are dealt with, and the basic RS232 type interface scheme for serial signals is explored. The following chapter looks at some of the special interface chips designed for implementing a serial input–output interface. This chapter also looks at the actual line signals and deals with both the RS232C and the newer RS422A and RS423A type signal schemes.

Counting and timing are activities which are frequently required in a microcomputer system. It is possible to implement both counting and timing functions by using a software approach, but this tends to be inefficient. It is better to make use of external hardware to perform these functions, thus allowing the processor to carry out other tasks. In Chapter 7 the basic techniques for interfacing simple counters and timers to a CPU are explored, and the methods for measuring time periods, signal frequency and pulse width are explained. The chapter goes on to look at special counter timer chips and a typical real time clock chip.

In the real world outside the microcomputer most of the signals and parameters being measured are analogue in form, whilst the signals within the CPU are all in digital format. To interface between the real world and the CPU, some form of conversion between the analogue and digital forms of signal is required. Chapter 8 looks at some of the techniques for producing analogue output, and the arrangements for interfacing digital to analogue converters to the CPU bus systems. In Chapter 9 the techniques for inputting analogue signals are examined with a look at the common types of analogue to digital converter and their basic characteristics.

In a microcomputer system that has to operate in real time, as for instance in a control system, it is important that the CPU should be able to respond immediately to hardware signals from the external world. Although this can be done by regularly checking the input lines using a software polling scheme, a more effective method is to make use of the interrupt facility which is provided on all modern general purpose microprocessors. The final chapter examines the interrupt facilities provided on the popular 8 bit microprocessors, and explores the basic principles of using hardware interrupts with a microcomputer to allow an external hardware signal to influence the program execution directly.

S. A. Money

Contents

Chapter One
CPUs and Bus Systems

The basic arrangement of virtually all current microcomputer systems follows the pattern shown in Figure 1.1. At the centre is the CPU (Central Processor Unit) which contains the complex logic that interprets the user's instructions and then controls the overall computer system to execute those instructions. The instructions themselves form what is called the program and consist simply of numbers which are stored in a memory unit. The memory unit is also used to hold data that is to be processed, intermediate answers and the final results of a computation. In any computer system, if it is to be of practical use, there must be some means of feeding data and instructions into the system and of outputting results. This is achieved by using a series of input and output ports which connect the CPU to the outside world. The CPU, memory and input–output circuits are tied together by three sets of wires, each of which is known as a bus.

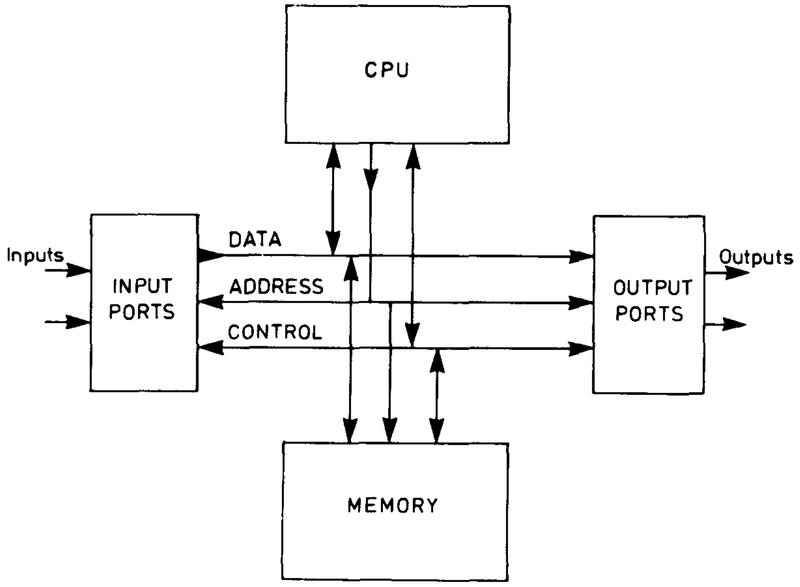


Fig. 1.1 Basic organisation of a microcomputer system.

The data bus is used to carry data between the CPU and the memory or the input–output channels. This bus also carries the instructions from the memory to the CPU so that they can be executed. The data bus is bi–directional allowing signals to be passed either to or from the CPU over the same set of wires. The CPU itself controls which direction data will flow across the data bus. When data flows from the CPU this is referred to as a 'write' operation whilst a data flow into the CPU is called a 'read' operation. Typical current CPU devices use either 8 or 16 wires for the data bus.

A second set of wires forms the *address* bus which provides a set of signals, known as the 'address' that is used to select a particular memory location so that data may be transferred between the CPU and that location. On the address bus data always flows from the CPU to the external device. Where there are several input and output channels in the system the address bus may also be used to select a particular channel for connection to the data bus. Most of the popular microprocessor devices use a 16 line address bus but some of the more complex types, such as the 8086 and 68000, use 20 or 24 address lines.

The third bus is a *control* bus which carries a number of control and timing signals out from the CPU and also allows control signals to be fed into the CPU. This bus is used to link the timing and operation of the various parts of the system so that the memory or input–output devices can be connected to the data bus at the appropriate times when the CPU is ready to carry out a data transfer. *Read* and *write* control signals from the CPU are used to indicate which way the data is to flow across the data bus wires and these signals allow the memory or I/O circuits to be set up either to receive data or to send it to the data bus. Other control signals are used to indicate when a valid address is available on the address bus and other signals may be provided to indicate whether the data transfer is to be made with the memory or an I/O channel.

In a computer system used for straightforward computation the input will generally be from a keyboard and output may be to a visual display screen or to some form of printer. In a computer that is used for control applications the inputs may be signals from transducers which are measuring real physical parameters on the process being controlled and the outputs will often be analogue signals which control the process equipment to achieve the required performance. Like the memory section the input and output ports are coupled to the CPU via the data and address buses.

The popular general purpose microprocessor chips, such as the 6809 and Z80 provide only the basic CPU function and will need to have memory and input–output ports added to them in order to produce a working system. The process of connecting these circuits to the computer is usually referred to as interfacing. In this book we shall be examining both the principles and practical aspects of providing the interfaces between the CPU and memory or input–output channels. Before going on to look at the actual process of interfacing devices to a CPU it might be as well at this point to examine the bus systems and timing arrangements of some of the more popular general purpose microprocessors.

Logic signals and devices

In all digital logic systems the actual signals used have only two basic states which are generally referred to as the 0 and 1 states and these are defined by the voltage level of the signal. Typically the 0 state is a signal with a value of less than +0.5 V and the 1 state is usually represented by a signal which has a voltage level of +2.5 V or more. All of the popular microprocessors work with a single +5 V power supply line and in most cases the actual voltage for a 1 level will be of the order +4 to +4.5 V. Other names for the 0 level are 'low' or 'false' whilst the 1 level may be referred to as a 'high' or 'true' state.

Since a digital signal on a single wire has only two possible states it can represent only the numbers 0 and 1. In order to deal with larger numbers the combination of states on a set of wires is used where the signal on each wire is referred to as a 'bit' and the combination of bit states on a set of wires is called a 'word'. Each bit in the word is allocated a weight which is a power of 2 so that working up from the least significant end of a word the bits represent units, twos, fours, eights and so on as shown in Figure 1.2.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit Ø
Numeric value	128	64	32	16	8	4	2	1

Fig. 1.2 Layout and bit values of a data word using the binary code system.

If we consider the typical 8 bit word passed across an 8 line data bus the most significant bit will have a value of 128 (2^7) and if all of the bits in the word were set at 1 the value of the complete word is 255. Thus for an 8 bit system numbers from 0 to 255 can be represented by a single data word. For each additional bit added to a data word the range of possible values is doubled, so for a 16 bit word numbers from 0 to 65 535 can be represented. This method of coding the logic data signals is referred to as pure binary code. Various other coding methods may be used for the basic binary data to suit the needs of the computer system and we shall look at some of these later in the book.

Various types of circuitry are used in making the CPU chips and the other logic chips that are needed to drive peripheral circuits around the CPU. Most of the current CPU chips use some form of NMOS (N channel Metal Oxide Semiconductor) technology for the fabrication of the chip. This provides fairly high speed operation with relatively low supply current but the output circuits are usually rather limited in the amount of current that they can handle. Another popular form of construction for the CPU is CMOS (Complementary Metal Oxide Semiconductor) which has the advantage of very low current consumption and high tolerance of power supply voltage variations.

For supporting logic devices the most popular type is the 74 series TTL (Transistor Transistor Logic) range of circuits. There are in fact several variations of the 74 series devices. The original standard 74 series logic circuits give quite high switching speed but require fairly high supply current so that a complex logic circuit can make very heavy demands on the power supply. A low power series with 74L type numbers is available in which the devices have a reduced supply current requirement but provide a much lower switching speed. In fact the 74L series devices are generally too slow in operation for use in modern microprocessor circuits. A more recent development is the 74S Schottky series which have similar characteristics to the standard 74 range but with much higher switching speed. A low power version of Schottky TTL, called the 74LS series, can be obtained and these circuits combine low power consumption with roughly the same operating speed as the standard 74 series. For most applications in modern microcomputer systems the 74LS series devices are generally used to provide the support and interface circuits needed around the CPU chip.

Another popular type of logic circuit is the CMOS variety which have type numbers in the 4000 series. The CMOS circuits have the advantage of extremely low power consumption and a relative insensitivity to power supply variations which makes them ideally suited for use in portable equipment which is operated from a battery supply. Some types of CPU are also available as CMOS versions. In general the basic 4000 series devices do not have a sufficiently high switching speed to operate properly with a modern CPU running at its maximum speed but faster versions of the CMOS circuits are available to meet the speed requirements of typical CPUs.

A TTL logic circuit in the 74 series has an input circuit which when pulled down to the 0 level provides 1.6 mA of current which must be absorbed by the driving circuit. This 'sink' current of 1.6 mA is called a standard TTL load and the normal 74 series output stages are designed to sink a current of 16 mA so that they can successfully drive up to 10 standard TTL input circuits. The LS versions are able to drive up to 5 standard TTL inputs. The input load for an LS type circuit is however only 0.4 mA so an LS type output will drive up to 20 LS type input circuits.

Logic output circuits

The conventional TTL logic gate or flip-flop circuit normally has a totem pole type output stage which actively drives the output to either the 0 or 1 logic level. A typical totem pole circuit is shown in Figure 1.3(a). If two such outputs are connected in parallel they would become mutually destructive if set to different levels with the result that one or both of the output transistors in each device could be destroyed. Where several circuits are to drive a common set of bus wires some alternative form of output stage is required to avoid this potentially catastrophic situation.

One solution is the open collector output circuit such as that shown in Figure 1.3(b). Here the upper transistor has been removed. To work properly this circuit requires an external pull up resistor which will complete the path between the output collector and the positive supply rail. When the output stage is turned off no current flows through the resistor and the voltage at the output terminal rises to the 1 level.

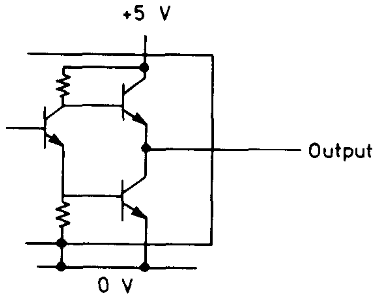
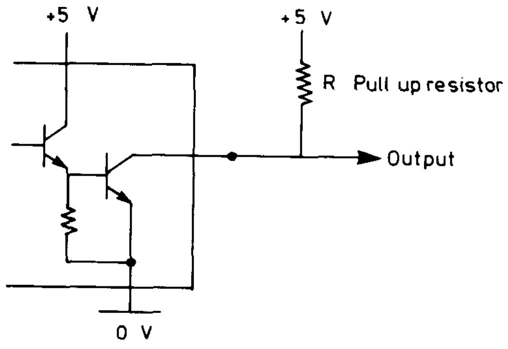


Fig. 1.3 (a) Totem pole type output stage.



(b) Wired OR type output circuit.

When the output stage turns on it draws current and pulls the output voltage down to the 0 level.

If a second open collector stage is connected in parallel it has no effect when its output stage is turned off but it also will pull the output line down to the 0 level when its output stage is turned on. When one stage pulls down the output it merely diverts current from any other stage that is turned on but cannot cause damage to the other output circuits connected to the common signal rail. This form of connection is called wired OR since the output falls to 0 if either one OR other of the circuits driving the common point goes to the 0 level.

With a wired OR system the value of the pull up resistor must be high enough so that one single output stage drawing current through the load is able to pull the output voltage down to the 0 level and still sink current for any other logic inputs that are to be driven. The load to ground which can be connected across such a circuit must have a sufficiently high resistance that when the circuit is at the 1 level the current drawn by the bus circuit does not pull down the logic level below the 1 state.

Although the wired OR configuration is convenient where a number of devices are driving a common data line this mode does have some disadvantages in terms of drive capability and speed of operation. An alternative output arrangement for logic devices is known as a tri–state output.

Normal logic devices have two basic output states which produce the 1 and 0 logic levels and actively drive the output line. In a tri-state output circuit the output stages can be disabled so that the output circuit effectively becomes an open circuit. This third disabled state allows other logic devices to drive the output data line with no danger of destruction of the output transistors. When the tri–state device is required to drive the output line its output circuit is enabled and once again it acts as a normal logic output circuit. Only one device must be allowed to drive the common line at any time, so when one device is enabled all other circuits driving that line must be disabled by switching them into their tri–state or open circuit mode. Tri–state logic circuits are almost universally used for driving the data bus of a microprocessor system with the state of each device determined by control signals from the CPU so that the correct device is enabled to the data bus according to the data transfer action called for by the CPU.

The Motorola 6800/6809 CPU

Let us start by examining the Motorola series 6800 microprocessors which have a straightforward and easy to understand bus system and timing sequence. This range of processors includes the original 6800, the 6802 and the 6809. The 6800 and 6802 are basically identical except that the 6802 has built in clock generation circuits and a slightly different pin layout. The 6809 is a more advanced version of the 6800 with more internal registers and an enhanced instruction set but its basic operation is similar to that of the 6800. Like the 6802 the basic 6809 has built in clock generation circuits although one version is available which uses an external clock in the same way as the 6800. The pin connections and signals for the 6802 are shown in Figure 1.4 whilst those of the 6809 are shown in Figure 1.5.

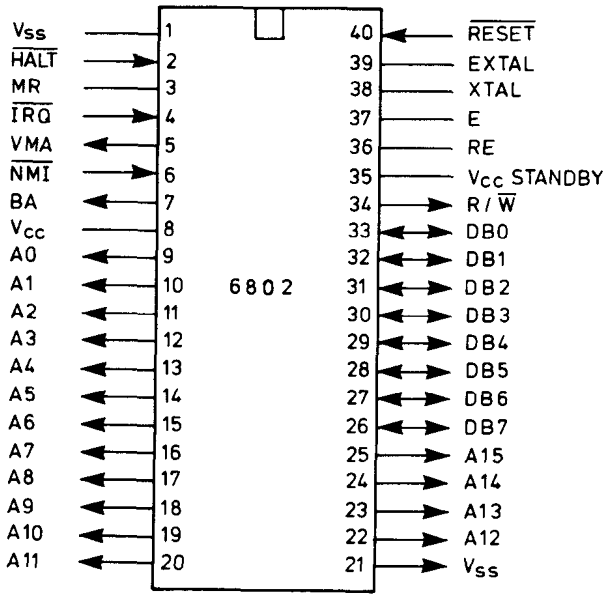


Fig. 1.4 Pin connections and signals for a 6802 CPU.

The 6802 operates internally with an 8 bit data word and uses an 8 bit data bus so it is generally classed as an 8 bit processor. The eight data bus lines DB0 to DB7 are bi-directional and the direction of flow across the data bus is governed by the R/W control signal which is output by the CPU. This R/W signal is at 1 for a read operation where data flows into the CPU and at 0 for a write operation where data flows to the memory or output channel. This R/W output from the CPU is normally at the 1 state but for operations which require the CPU to write data onto the data bus the R/W line goes to 0 during the phase 2 clock cycle before the data transfer occurs.

The 6802 and 6809 both use a fully synchronous bus system with its timing based on a simple two phase CPU clock and its operation is relatively easy to understand. The two phases of the clock signal are in

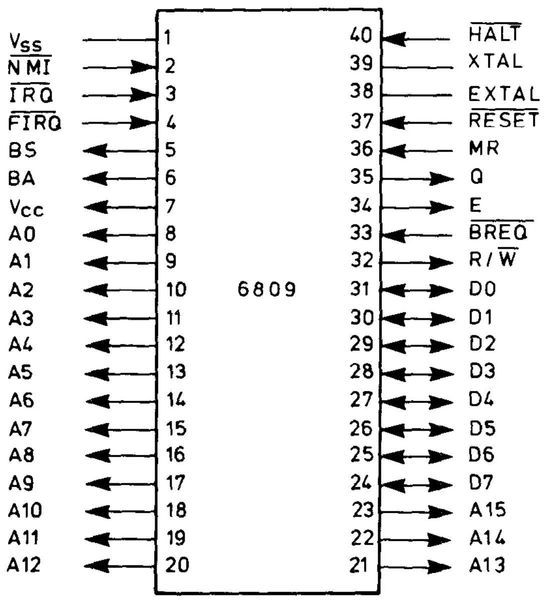


Fig. 1.5 Pin connections and signals for a 6809 CPU.

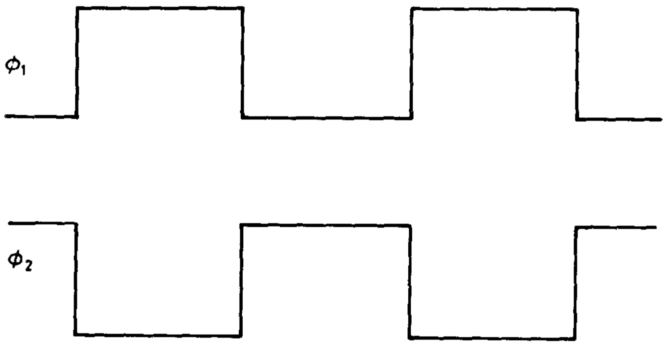


Fig. 1.6 The two phase clock of the 6802 processor.

antiphase and timed so that they do not overlap as shown in Figure 1.6. Phase 1 (ϕ1) is used for operations within the CPU itself and is not normally required by the external devices in the system. All data transfers to or from the CPU via the data bus are made during the period when the phase 2 (ϕ2) clock is at the 1 level. The ϕ2 clock is therefore used to time data transfers to or from the memory or the input–output ports. The standard version of the 6802 runs at a clock rate of 1 MHz so each clock takes 1 microsecond. Since most instructions require some 2 to 4 clock cycles the program execution speed is around 300 000 to 400 000 instructions per second. Faster versions such as the 68A02 and 68B02 will run at clock rates of 1.5 MHz and 2 MHz respectively. The 6809 CPUs run at the same speed and there are also A and B versions of the 6809 for higher speed operation.

The sequence of events as each program instruction is executed consists of a number of machine cycles. Typical instructions will take from 2 to 5 of these cycles for execution. The first cycle of any instruction is always an opcode fetch which reads in the opcode data from memory and the last cycle is where the operation specified by the instruction takes place. Some instructions such as INY (Increment Y register) only require the opcode so execution takes place during the machine cycle following the opcode fetch. Other instructions require an operand, such as an address, and will include one or two extra machine cycles during which the operand data is read in from memory before the final cycle of the instruction when execution takes place. In the 6800 series processors each machine cycle takes up one clock cycle of the CPU clock. Figure 1.7 shows the sequence for an instruction which stores the contents of the accumulator in memory.

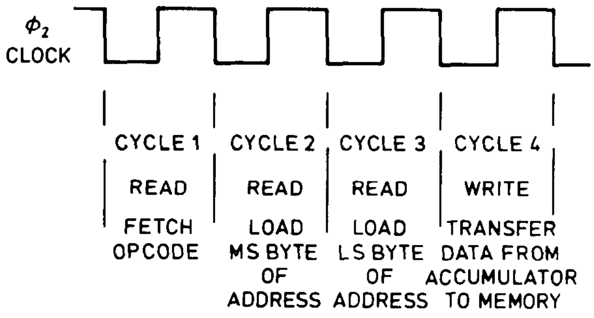


Fig. 1.7 Machine cycle and clock timing for a 6802 instruction execution sequence.

The address bus is 16 bits wide and is output on line A0 to A15 allowing 65 536 possible memory locations to be addressed. This bus is driven from one of the registers in the CPU and for proper operation the signals on the address lines must be stable and represent a valid address before data is transferred across the data bus. To achieve this the CPU outputs a VMA (Valid Memory Address) signal when a correct address is present on the address bus. This VMA signal is normally at 0 but goes to 1 when the address is set up ready for a data transfer. Both VMA and R/W signals are set up before the start of $\phi 2$ of the clock cycle.

The drivers within the CPU for writing to the data bus are normally only enabled during an instruction that calls for data to be written to the bus and at other times the drivers are disabled. Sometimes it may be desirable that these drivers should be disabled to allow another device to drive the bus and this can be done by setting the Data Bus

Enable (DBE) input line to the CPU to the 0 state. In most systems this DBE line is simply tied to the ϕ2 clock so that the drivers are enabled only when a write operation might be expected to take place.

For applications such as DMA (Direct Memory Access) or multi-processor operation it is desirable that the CPU should be able to release the address and data buses to allow another device to take control of them. This can be achieved on the 6800 series processors by applying a 1 input to the TSC (Tri–State Control) input which effectively disables the W/R control and address output lines. The data bus can be disabled by setting the DBE line low at the same time. When the bus is released an output signal BA (Bus Available) goes high and may be used to signal the external device that it may take control of the bus system. The operation of the CPU can be stopped by setting the HALT input at 0 which will also place the data and address buses in the tri–state condition.

The IRQ and NMI lines, and FIRQ on the 6809, are interrupt inputs which enable actions within the CPU to be triggered by external hardware signals. We shall be looking in more detail at the operation of these interrupt signals in a later chapter. The RESET input is also a form of interrupt signal which is used to reset all of the internal circuits of the CPU and start the program executing. The RESET line is set at 0 for a short period after power has been applied to the system and resets the internal circuits of the CPU. After this reset sequence the RESET input should be set at 1 for normal operation of the CPU.

Rockwell 6500 series CPU bus

The Rockwell 6500 series of CPUs, originally developed by MOS Technology, are 8 bit processors which use a similar design philosophy to that of the Motorola 6800 series. The internal architecture is slightly different and the instruction set although simpler provides one or two more versatile operations. Although there are a number of different CPUs in this series the most widely used is the 6502 which is fitted in many of the popular personal microcomputer systems such as the Apple series. The pin layout and signals for the 6502 are shown in Figure 1.8.

It should be noted that unlike the 6800 series, this CPU does not produce a VMA (Valid Memory Address) signal on its control bus. Address data will normally be valid from about 110 microseconds after the low to high transition of the ϕ1 clock until the end of the ϕ2 clock period. An extra signal that is provided is the SYNC signal which is set at 1 when an instruction fetch operation is being carried out. The READY input on the 6502 performs much the same function as HALT on the 6800 and is used to temporarily halt the CPU operation.