OVERVIEW

The SA828/SA838 Pulse Width Modulation ICs incorporate a MOTEL interface which primarily allows them to be controlled by either Intel or Motorola microprocessors and microcontrollers employing an 8-bit multiplexed address/data bus. However, since the SA8X8 is implemented using fully static logic, the bus can be controlled at lower speeds, making it suitable for use with microcontrollers which have no external bus.

This facility has become increasingly important as the cost of such microcontrollers has fallen and on-chip peripheral integration levels have risen. Microcontrollers such as (but not limited to) the Microchip PIC, Philips 87C75X, Zilog Z86, SGS Thomson ST6 and National Semiconductor COP† all feature the maximum number of I/O ports in any given package size by dispensing with external address/data busses and instead placing modestly sized RAM, EPROM (usually one-time programmable), and in some instances E2PROM, on-chip. Additionally, these devices often benefit from on-chip analogue to digital converters, timer/counters and interrupt handlers.

A corresponding decrease in price is evident in the power electronics which partners the SA8X8 and low-cost microcontroller, with the result that three phase induction motor control is now possible in price-critical applications such as consumer goods. In particular, products such as washing machines, domestic heating pumps and air conditioning units are now candidates for such control schemes, where the benefits of increased motor longevity and improved control may be realised.

This applications note details a means of connecting each of the microcontrollers listed above to the SA8X8 devices, and a software routine emulates a multiplexed address/data bus for each. The profusion of variants within each microcontroller family prevents any detailed discussion of how the various on-chip features may be used in a motor control application. This will be dealt with in subsequent applications notes.

ADDRESS/DATA BUS SCHEME

Figs. 1a and 1b show the detailed timing diagrams for both Intel and Motorola modes of operation. In each case it is assumed that the SA8X8 device is one of many peripheral devices on the bus and therefore a chip select signal (CS) is used to strobe between them.

Clearly, it is irrelevant which of these two forms is emulated by the microcontroller port pins, since both perform exactly the same function. In practice, however, Intel mode has the advantage that the read signal (RD) remains high for the duration of the cycle - since the SA8X8 is a write-only device. As a result, the WR pin may be tied permanently high.

This leaves three control lines - address latch enable (ALE), write (WR) and chip select (CS). If the SA8X8 is the only device on the emulated bus, CS is not required and may be tied permanently low.

The timing of the various signals may be split into five essential sections:

1. CS low, ALE high
2. Address set up time
3. ALE low, address hold time
4. Data setup time, WR low
5. Data hold time, WR high, CS high

This applications note details a means of connecting each of the microcontrollers listed above to the SA8X8 devices, and a software routine emulates a multiplexed address/data bus for each. The profusion of variants within each microcontroller family prevents any detailed discussion of how the various on-chip features may be used in a motor control application. This will be dealt with in subsequent applications notes.

ADDRESS/DATA BUS SCHEME

Figs. 1a and 1b show the detailed timing diagrams for both Intel and Motorola modes of operation. In each case it is assumed that the SA8X8 device is one of many peripheral devices on the bus and therefore a chip select signal (CS) is used to strobe between them.

Clearly, it is irrelevant which of these two forms is emulated by the microcontroller port pins, since both perform exactly the same function. In practice, however, Intel mode has the advantage that the read signal (RD) remains high for the duration of the cycle - since the SA8X8 is a write-only device. As a result, the WR pin may be tied permanently high.

This leaves three control lines - address latch enable (ALE), write (WR) and chip select (CS). If the SA8X8 is the only device on the emulated bus, CS is not required and may be tied permanently low.

The timing of the various signals may be split into five essential sections:

1. CS low, ALE high
2. Address set up time
3. ALE low, address hold time
4. Data setup time, WR low
5. Data hold time, WR high, CS high
Whilst the minimum timings given in Table 1a must be adhered to, in practice this does not present a problem since the instruction cycle time of most microcontrollers is of the order of 1µs. The time between any of the five sections listed above is constrained to be at least one instruction but is not subject to a maximum due to the static nature of the SA8X8 MOTEL interface.

The timing diagram may be redrawn as shown in Fig.2. Note that ALE is permitted to go high at any time, provided that t8 (chip select setup) and t4 (delay from WR high to CS low) times are adhered to. Hence, chip select may be exerted at the same time as ALE goes high, so long as WR from a previous instruction goes high at least t4 before this. In fact this is guaranteed due to period 5 in Fig.2. Similarly, period 5 ensures that the t9 (chip select hold) period is adhered to.

Note that there is no necessity to place the address/data lines into a quiescent state after WR becomes inactive since the next instruction will set up the address before the ALE low-going edge. This is guaranteed to occur more than a t12 period following the WR rising edge by virtue of periods 1 and 5 in Fig.2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min.</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALE high period</td>
<td>t₁</td>
<td>70</td>
<td>ns</td>
</tr>
<tr>
<td>Delay time, ALE to WR</td>
<td>t₂</td>
<td>40</td>
<td>ns</td>
</tr>
<tr>
<td>WR low period</td>
<td>t₃</td>
<td>200</td>
<td>ns</td>
</tr>
<tr>
<td>Delay time, WR high to CS low</td>
<td>t₄</td>
<td>40</td>
<td>ns</td>
</tr>
<tr>
<td>CS setup time</td>
<td>t₅</td>
<td>20</td>
<td>ns</td>
</tr>
<tr>
<td>CS hold time</td>
<td>t₆</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>Address setup time</td>
<td>t₁₀</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>Address hold time</td>
<td>t₁₅</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>Data setup time</td>
<td>t₁₁</td>
<td>100</td>
<td>ns</td>
</tr>
<tr>
<td>Data hold time</td>
<td>t₁₂</td>
<td>25</td>
<td>ns</td>
</tr>
</tbody>
</table>

Table 1a Intel bus timings at $V_{DD} = 5V$, $T_{AMB} = +25^\circ C$

This may be translated into a flow diagram, as shown in Fig.3:

![Fig.2: Address/data bus timing diagram (Intel Mode)](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min.</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>AS high period</td>
<td>t₁</td>
<td>90</td>
<td>ns</td>
</tr>
<tr>
<td>Delay time, as low to DS high</td>
<td>t₂</td>
<td>40</td>
<td>ns</td>
</tr>
<tr>
<td>DS high period</td>
<td>t₃</td>
<td>210</td>
<td>ns</td>
</tr>
<tr>
<td>Delay time, DS low to AS high</td>
<td>t₄</td>
<td>40</td>
<td>ns</td>
</tr>
<tr>
<td>DS low period</td>
<td>t₅</td>
<td>200</td>
<td>ns</td>
</tr>
<tr>
<td>DS high to R/W low setup time</td>
<td>t₆</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>R/W hold time</td>
<td>t₇</td>
<td>10</td>
<td>ns</td>
</tr>
<tr>
<td>CS setup time</td>
<td>t₈</td>
<td>20</td>
<td>ns</td>
</tr>
<tr>
<td>CS hold time</td>
<td>t₉</td>
<td>0</td>
<td>ns</td>
</tr>
<tr>
<td>Address setup time</td>
<td>t₁₀</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>Address hold time</td>
<td>t₁₅</td>
<td>30</td>
<td>ns</td>
</tr>
<tr>
<td>Write data setup time</td>
<td>t₁₁</td>
<td>110</td>
<td>ns</td>
</tr>
<tr>
<td>Write data hold time</td>
<td>t₁₂</td>
<td>30</td>
<td>ns</td>
</tr>
</tbody>
</table>

Table 1b Motorola bus timings at $V_{DD} = 5V$, $T_{AMB} = +25^\circ C$

**IMPLEMENTATIONS**

The following sections detail the interconnection diagrams between each of the chosen microcontrollers and the SA8X8 and a subroutine which implements the flow diagram of Fig. 3. For the purposes of direct comparison, the same names have been used for variables and pins in each example. Specific port pins have been selected for each interconnection purely for convention. These may be changed simply by redefining the equates in the source code headers.
1. MICROCHIP PIC SERIES.

The port pins of the PIC microcontrollers are bi-directional and must be set up as inputs or outputs in the first few lines of code. When defined as an output each port pin is capable of sourcing or sinking 25mA. It is therefore very important that an input cannot be inadvertently redefined as an output since this could cause contention and possibly destroy the device. For this reason the port initialisation routine has been made a subroutine and should be called regularly from the body of the code.

;************************************************************************
;BYTE EQUATES
;************************************************************************
PORT_A EQU 5 ;PHYSICAL LOCATIONS OF PORTS A & B
PORT_B EQU 6
ADDRESS EQU 10 ;ADDRESS AND DATA REGISTERS
DATA EQU 11

;************************************************************************
;BIT EQUATES
;************************************************************************
ALE EQU 0 ;PORT_A BIT 0= ALE
WRB EQU 1 ;PORT_A BIT 1= WRB
CSB EQU 2 ;PORT_A BIT 2= CSB

;************************************************************************
PORT_INITIALISE ;SETS UP QUIESCENT STATE OF PORTS. CALL
;DURING INITIALISATION ONLY
;************************************************************************
BSF PORT_A,CSB ;QUIESCENT IS CSB HIGH, WRB HIGH
BCF PORT_A,ALE ;AND ALE LOW.
BSF PORT_A,WRB
MOVLW 0FFH
MOVWF PORT_B
RETLW 00H

;************************************************************************
PORT_DEFINITION_ROUTINE ;SETS UP I/O PORTS AS INPUTS OR OUTPUTS
;CALL DURING INITILISATION AND
;REGULARLY FROM BODY OF PROGRAM
;TO PREVENT CONTENTION
;************************************************************************
MOVLW XXXXX000B ;3 LSB'S OF PORT_A ARE O/PS
TRIS PORT_A
MOVLW 00H ;PORT_B ALL OUTPUTS
TRIS PORT_B
RETLW 00H

;************************************************************************
BUS_EMULATOR ;IMPLEMENTS SECTIONS 1-5 OF FIGURE 3
;IMPORT ADDRESS AND DATA WORDS
;************************************************************************
BCF PORT_A,CSB ;SECTION 1. ALE HIGH, CSB LOW
BSF PORT_A,ALE
MOVF ADDRESS,W ;SECTION 2. SET UP ADDRESS ON
MOVWF PORT_B ;AD0-7
BCF PORT_A,ALE ;SECTION 3. ALE LOW
BCF PORT_A,WRB ;SECTION 4. WRB LOW AND
MOVF DATA,W ;SETUP DATA TO AD0-7
MOVWF PORT_B
BSF PORT_A,WRB ;SECTION 5. WRB & CSB HIGH
BSF PORT_A,CSB
RETLW 00H ;RETURN

Fig.4: Microchip PIC16CXX interconnection diagram
2. PHILIPS 87C75X SERIES

The port pins on these devices, like all MCS51 derivatives, are “quasi bi-directional”. This means that they have weak pull-up resistors and a single transistor which when turned on will take the output low. Therefore, to configure such a port as a high output, ensure that the transistor is off and to configure as a low output ensure that it is on. To use this port as an input, first ensure that the transistor is off and then read the pin in the usual way. Because the pull-up resistor is weak the pin may pulled low externally.

Consequently, this range of microcontrollers does not have direction registers. The only requirement is to ensure that the transistor is off when using any given port as an input. (NOTE: Some ports are open-drain only. Please check when reconfiguring functions to different port pins than those specified).

---

**Fig.5: Philips 87C75X interconnection diagram**

---

```assembly
;************************************************************************
;INTERNAL RAM EQUATES
;************************************************************************
ADDRESS DATA 08H ;ADDRESS AND DATA REGISTERS
DATA DATA 09H ;PLACED ABOVE REG BANK 0

;************************************************************************
;BIT EQUATES
;************************************************************************
ALE BIT P3.0 ;PORT 3 BIT 0= ALE
WRB BIT P3.1 ;PORT 3 BIT 1= WRB
CSB BIT P3.2 ;PORT 3 BIT 2= CSB

;************************************************************************
PORT_INITIALISE ;SETS UP QUIESCENT STATE OF PORTS. CALL
;DURING INITIALISATION ONLY

           SETB CSB ;QUIESCENT IS CSB HIGH, WRB HIGH
           CLR ALE ;AND ALE LOW.
           SETB WRB
           MOV P1,#0FFH
           RET

;************************************************************************
BUS_EMULATOR ;IMPLEMENTS SECTIONS 1-5 OF FIGURE 3
;IMPORT ADDRESS AND DATA WORDS
;************************************************************************
           CLR CSB ;SECTION 1. ALE HIGH, CSB LOW
           SETB ALE
           MOV P1,ADDRESS ;SECTION 2. SET UP ADDRESS
           CLR ALE ;SECTION 3. ALE LOW
           CLR WRB ;SECTION 4. WRB LOW AND
           MOV P1,DATA ;SETUP DATA TO A0-7
           SETB WRB ;SECTION 5. WRB & CSB HIGH
           SETB CSB ;RETURN
```

---

AN4677
The port pins on these devices are variously configured as I/O, inputs or outputs. Taking the One-Time-Programmable Z86E04 as an example, P0 is a three bit I/O port, P1 is a three bit input only port and P2 is an eight bit I/O port. For simplicity the following example uses P2 as the address/data bus and P0 as the control bus (CS, WR and ALE).

The instruction set makes no provisions for ‘bit set’ and ‘bit clear’ instructions to toggle port pins high and low. However, a read-modify-write architecture allows bit setting, clearing and toggling using OR, AND and XOR functions respectively. Since the initial read operation reads the state of the port pins themselves, rather than the port data registers, there is a risk of erroneous operation if the two states do not agree. This may occur particularly if the ports are defined to be open-drain and there is no external means for pulling the pin high. For this reason all the port pins in the listings below have been set to push-pull mode. (An additional safeguard consists of keeping an ‘image’ of the port data in a separate data register and then writing it to the port data registers after each change).

In this particular case, however, using P0 as the three-bit control bus means that it is never necessary to set or clear individual port pins. It is adequate simply to write to all three pins simultaneously since they are all controlled exclusively inside the BUS_EMULATOR routine.

```
;************************************************************************
;INTERNAL RAM EQUATES, REGISTER BANK 0
;************************************************************************
ADDRESS .equ 04H ;ADDRESS AND DATA REGISTERS
address .equ r4 ;PLACED ABOVE PORTS IN REG BANK0
DATA .equ 05H
data .equ r5

;************************************************************************
PORT_INITIALISE ;SETS UP QUIESCENT STATE OF PORTS.
;************************************************************************
LD P0,#110B ;CS/ HIGH, WR/ HIGH, ALE LOW
LD P2,#%FF ;SET ADDRESS/DATA BUS TO QUIESCENT
LD P2M,#%00 ;PORT2 ALL OUTPUTS
LD P3M,#%01 ;P3 I/PS DIGITAL, P2 OP/S PUSH/PULL
LD P01M,#%04 ;PORT0 ALL OUTPUTS
RET

;************************************************************************
BUS_EMULATOR ;IMPLEMENTS SECTIONS 1-5 OF FIGURE 3
;IMPORT ADDRESS AND DATA WORDS
;************************************************************************
LD P0,#011B ;SECTION 1. ALE HIGH, CSB LOW
MOV P2,ADDRESS ;SECTION 2. SET UP ADDRESS
LD P0,#010B ;SECTION 3. ALE LOW
LD P0,#000B ;SECTION 4. WRB LOW AND
MOV P2,DATA ;SETUP DATA TO AD0-7
LD P0,#110B ;SECTION 5. WRB & CSB HIGH
RET
```
4. SGS THOMSON ST6 SERIES.

The port pins of these devices are particularly versatile—allowing virtually any pin to be input, output (push-pull or open drain, internal pull-up optional), interrupts and in some cases analogue to digital converter inputs. For the reasons given in Section 3, all outputs have been defined as push-pull in this instance.

Three registers exist for each port: Option Register (ORx), Data Direction Register (DDRx) and Data Register (DRx). Individual bits of the DDR register defines whether the port is input (0) or output (1), whilst each bit of the OP register defines whether the pin is push-pull or open-drain (when configured as an output).

For a push-pull output both the DDR and OR bits need to be high. The DR register then holds the output data. For the purposes of this example, Port A has been used as the control bus and Port B as the address/data bus.

---

```asm
;********************************************************************
;INTERNAL RAM EQUATES, REGISTER BANK 0
;********************************************************************
ADDRESS .def 84H ;ADDRESS AND DATA REGISTERS
DATA .def 85H

ALE .equ 0 ;PORT B BITS EQUATES
WRB .equ 1
ALE .equ 2

;********************************************************************
PORT_INITIALISE ;SETS UP QUIESCENT STATE OF PORTS.
;********************************************************************
LDI DRB,0FFH ;SET ADDRESS/DATA BUS TO QUIESCENT
LDI DRA,X110B ;CS/ HIGH, WR/ HIGH, ALE LOW
LDI ORA,#0FH ;PORTA AND B SET TO PUSH-PULL
LDI ORB,#0FFH
LDI DDRA,#0FH ;PORTA AND B SET TO OUTPUTS
LDI DDRB,#0FFH
RET

;********************************************************************
BUS_EMULATOR ;IMPLEMENT SECTIONS 1-5 OF FIGURE 3
;IMPORT ADDRESS AND DATA WORDS
;********************************************************************
RES CSB,DRA ;SECTION 1. ALE HIGH, CSB LOW
SET ALE,DRA
LD A,ADDRESS ;SECTION 2. SET UP ADDRESS
LD DRB,A
RES ALE,DRA ;SECTION 3. ALE LOW
RES WRB,DRA ;SECTION 4. WRB LOW AND
LD A,DATA ;SETUP DATA TO AD0-7
LD DRB,A
SET DRB,DRA ;SECTION 5. AND CSB HIGH
SET CSB,DRA
RET ;RETURN
```

---

Fig. 7: SGS Thomson ST62 interconnection diagram
5. NATIONAL SEMICONDUCTOR COP SERIES.

The port pins of this range of microcontrollers differ substantially from one to the next. Port L, however, exists on all devices and is a general purpose eight-bit I/O port. In addition Port G exists on all devices although it is of variable length, dependent upon the size (pin-count) of the particular variant. Port L has therefore been adopted as the address/data port and Port G as the control port.

Two registers exist for each port: Port Configuration (PxC) and Port Data (PxD). Individual bits of the PxC register defines whether the port is input (1) or output (0), whilst each bit of the PxD register defines the output state.

---

; ****************************
; INTERNAL RAM EQUATES, REGISTER BANK 0
; ****************************
ADDRESS .def 84H ;ADDRESS AND DATA REGISTERS
DATA .def 85H

ALE .equ 0 ;PORT B BITS EQUATES
WRB .equ 1
ALE .equ 2

; ****************************
PORT_INITIALISE ;SETS UP QUIESCENT STATE OF PORTS.
; ****************************
SBIT CSB,PGD ;CS/ HIGH, WR/ HIGH, ALE LOW
SBIT WRB,PGD
RBIT ALE,PGD
LD PLD,#0FFH ;SET ADDRESS/DATA BUS TO QUIESCENT
LD PLC,#0 ;PORT L ALL OUTPUTS
LD PGC,#0 ;PORT G ALL OUTPUTS
RET

; ****************************
BUS_EMULATOR ;IMPLEMENTS SECTIONS 1-5 OF FIGURE 3
; IMPORT ADDRESS AND DATA WORDS
; ****************************
RBIT CSB,PGD ;SECTION 1. ALE HIGH, CSB LOW
SBIT ALE,PGD
LD A,ADDRESS ;SECTION 2. SET UP ADDRESS
LD PLD,A
RBIT ALE,PGD ;SECTION 3. ALE LOW
RBIT WRB,PGD ;SECTION 4. WRB LOW AND
LD A,DATA ;SETUP DATA TO ADD-7
LD PLD,A
SET WRB,PGD ;SECTION 5. WRB & CSB HIGH
SET CSB,PGD
RET ;RETURN
---

Fig.8: National Semiconductor COP interconnection diagram