INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION
INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
I.C.T.P., P.O. BOX 586, 34100 TRIESTE, ITALY. CABLE: CENTRATOM TRIESTE

H4.SMR. 403/19

FIFTH COLLEGE ON MICROPROCESSORS: TECHNOLOGY AND APPLICATIONS IN PHYSICS

2 - 27 October 1989

Microprocessor Interfacing

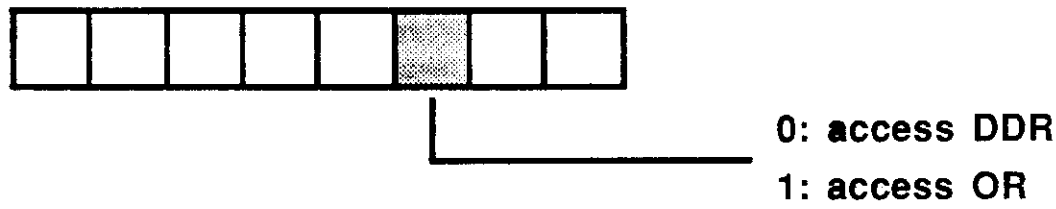
A. COLAVITA
Microprocessors Laboratory, ICTP, Trieste

Lecture 3: The PIA MC6821 - Part 2

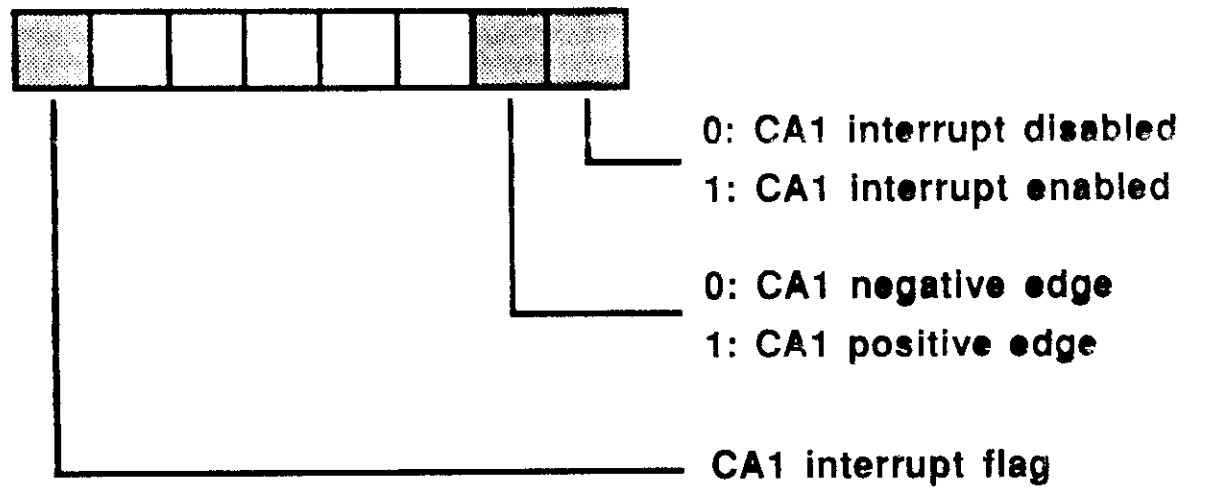
- Programming the Control Registers
- Using the PIA Control Lines
  - Count Contact Closures
  - Detect Contact Closure and Opening
  - Use CB2 as Output Line
  - Interface to Registers on a 4-bit Bus
  - Parallel 16-bit Data Link

1. Programming the Control Registers

In the last lecture we only looked at one bit in the control registers, the bit CRx-2, which is used to select either DDRx or ORx when addressing the PIA.



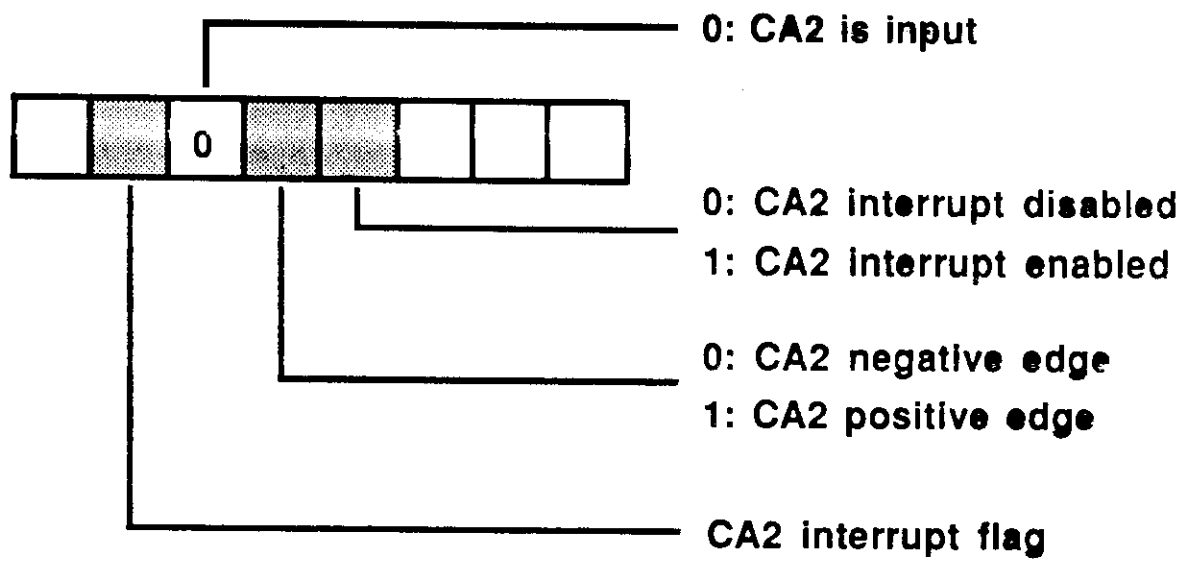
The remaining bits are used to control CA1, CA2, CB1 and CB2. We discuss the programming for the A side.



CA1 is an edge sensitive input line, i.e. it detects transitions rather than levels. CRA-1 is used to select the desired direction of the transition,

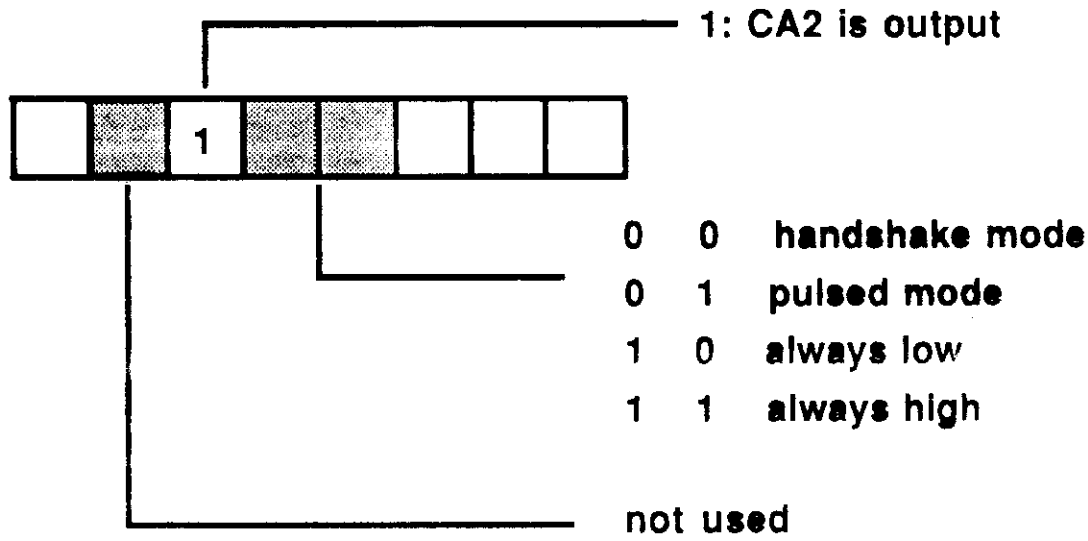
high-to-low or low-to-high. Once a transition has occurred, the CRA-7 bit is set and remains set even if the CA1 line changes state. This bit is read-only, writing to CRA will not affect it. The only way to reset this bit to zero is to read ORA even if all peripheral lines are programmed as outputs. Bits CRA-1 and CRA-7 are needed to use CA1 in programmed mode. CRA-0 is the interrupt enable bit and if set to 1, the IRQA* line is activated as soon as CRA-7 goes to 1.

CA2 can either be an edge sensitive input such as CA1or be used as output. CRA-5 selects on the direction of CA2.

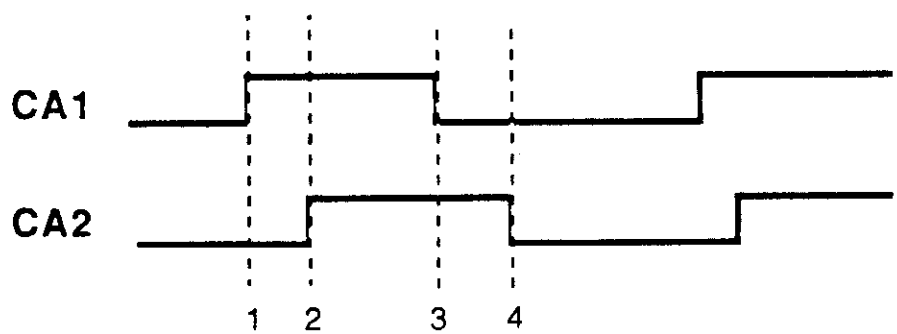


In input mode CRA-3, CRA-4 and CRA-6 have the same function for CA2 as CRA-0, CRA-1 and CRA-7 resp. for CA1. Clearing CRA-6 is also done by reading ORA. **Warning:** Reading ORA clears both flags !!

Setting CRA-5 to 1 switches CA2 to output mode. The interrupt flag bit CRA-6 is not used in this case. CRA-3 and CRA-4 allow for four different operating modes, which are handshake, pulsed, always low and always high. In handshake mode CA1 and CA2 are used together, in pulsed mode CA2 may be used alone or together with CA1 and in the last two cases CA2 is used like a normal output line.



Assume that CA1 is programmed for a low-to-high transition, that CA2 is in handshake mode and that data will come in from an external device. The hanshaking sequence is then as follows:



At the beginning, CA1and CA2 are low. The external device will put the data on the PA lines and rise CA1(1) which automatically turns CA2 high (2) and sets CRA-7. The device should turn CA1 low (3) but leave the data stable. The program would check CRA-7 periodically to discover the transition and read the data from ORA. This read operation clears CRA-7 and turns CA2 low (4).

In pulsed mode CA2 is normally high. Reading data from ORA produces a low going pulse with a width of one E cycle (1 µs for a 1MHz system).

For the B side all operations are the same with one exception: Output changes on CB2 occur after a write operation to ORB. Clearing of CRB-7 is still achieved by reading ORB.

2. Using the PIA Control Lines

The PIA control lines are used to synchronise the data transfer to and from the external equipment. They can be used in many different ways.

2.1 Count contact closures

Any of the four lines can be used to detect transitions. Assuming that the PIA was already initialised the following routine could be used to count transitions on CB1:

```
LOOP    TST     PIA+3           IF CRB-7 set
        BPL     NEXT
        TST     PIA+2           THEN    clear CRB-7
        INC     COUNT                   COUNT=COUNT+1
NEXT    BRA     LOOP
```

2.2 Detect contact closure and opening

CA1 is used to detect the closure and opening of a debounced switch. The closure will generate a high-to-low transition, the opening a low-to-high transition.

The following routine will do the job:

```
INIT    LDA     #4          Init PIA
        STA     CRA
        TST     ORA         Clear any pending flag
*
DOWN    TST     CRA         Wait for closure
        BPL     DOWN
        TST     ORA         Clear CRA-7 flag
        LDA     #6          select new transition
        STA     CRA
*
*       Do something else
*
UP      TST     CRA         Wait for opening
        BPL     UP
        TST     ORA         Clear CRA-7 flag
*
*       ... and so on ...
*
```

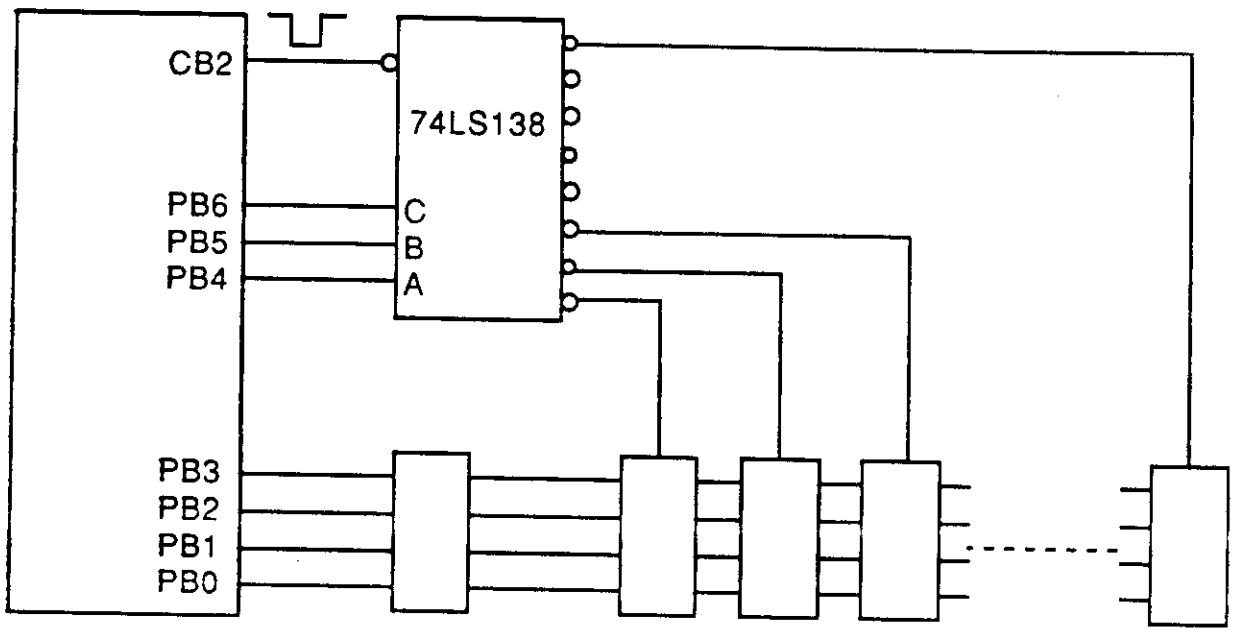
2.3 Use CB2 as output line

The following programming example shows how to use CB2 without affecting the programming of CB1 on the same PIA. This technique should be used in cases where different routines have to share one PIA port.

```
LOW     LDA     CRB         read present settings
        ANDA    #$C7        remove bits for CB2
        ORA     #$30        CB2 output and low
        STA     CRB         update PIA
*
HIGH    LDA     CRB         read present settings
        ORA     #$38        CB2 output and high
        STA     CRB         update PIA
```

2.4 Interfacing to 8 registers with a 4-bit bus

In this example side B of the PIA is used to interface to an external 4-bit output bus to connect to 8 4-bit registers. PB0 to PB3 act as data lines, Pb4 to PB6 as address lines and CB2 as timing signal.



This system can be easily expanded to 16 registers using PB7 as additional address line and adding another decoder. In applications where an 8-bit bus is needed, the second half of the PIA could be used or one of the external registers may be defined as data buffer for the upper 4 data bits.

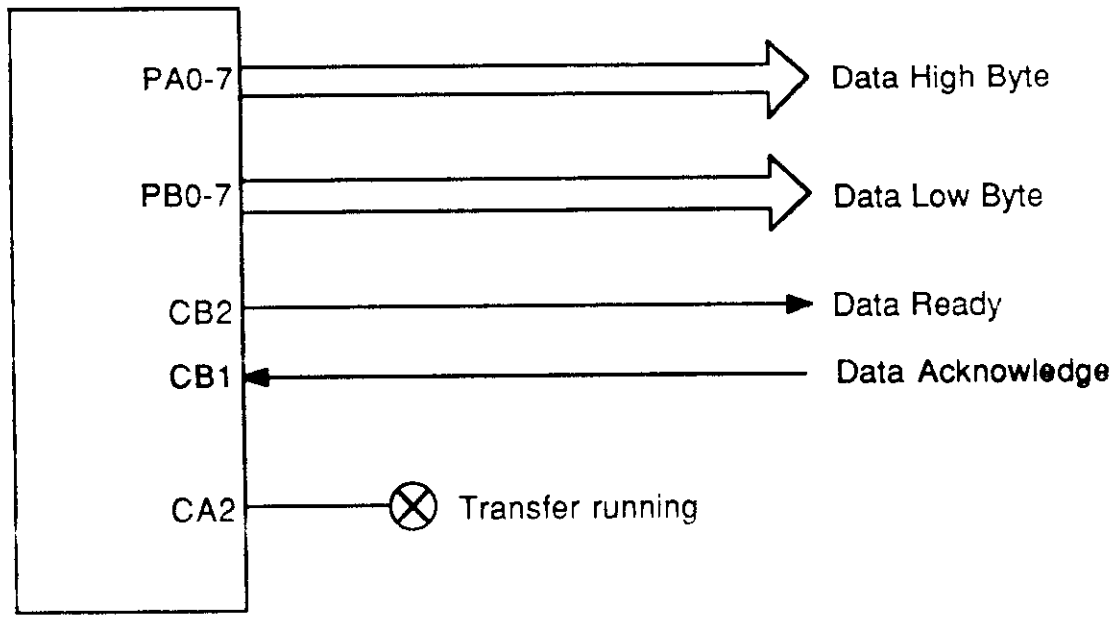
The following programming example shows how to transfer data from an array in memory to the 4-bit registers. It assumes that the 4-bit data words are stored in the lower half in an array of 8 bytes. The program combines the 4-bit data with a 3-bit address and stores these bytes in the PIA. The hardware will generate the CB2 strobe signal once the data are stable. Refer to the MC6821 data sheet for precise timing information.

```
*
*       Program to transfer a data array to a set of registers
*       The data are assumed to be in an array of 8 bytes in
*       the lower 4 bits, the higher 4 bits being 0.
*
PIA     EQU     $EF08           define PIA addresses
DDRB    EQU     PIA+2
ORB     EQU     PIA+2
CRB     EQU     PIA+3
*
        ORG     $1000
*
START   EQU     *               first init the PIA
        CLR     CRB             access DDRB
        LDA     #$FF            all lines output
        STA     DDRB
        LDA     #$2C            access ORB, CB2 pulsed mode
        STA     CRB
```

```
*
*       now copy array
*
COPY    LEAX    ARRAY,PCR       point to data
        CLRB                    array index and reg. address
NEXT    TFR     B,A             take address
        LSLA                    and move it up
        LSLA
        LSLA
        LSLA
        ORA     ,X+             join data and move pointer
        STA     ORB             move data out
        INCB                    next element
        CMPB    #8              all done?
        BLO     NEXT            no, take one more
*
*       continue ...
*
*
*       Static data array definition
*
ARRAY   FCB     1,2,3,4,5,6,7,8
        END     START
```

2.5 A parallel 16-bit data link

This example shows a parallel data link for high-speed data transfers between two microprocessors or between a microprocessor and a mini-computer. It is a unidirectional data transfer with handshaking protocol.



Programming is straight forward. First, both parts of the PIA are set up for output, CRA is set with CA2 as output and always low, CRB is set for handshaking. CB1 has to be programmed depending on the polarity generated by the receiving machine. If the polarity of CB2 is wrong there are two posiibilities: either one uses an inverter or the handshaking can be done under software control which is of course slower.