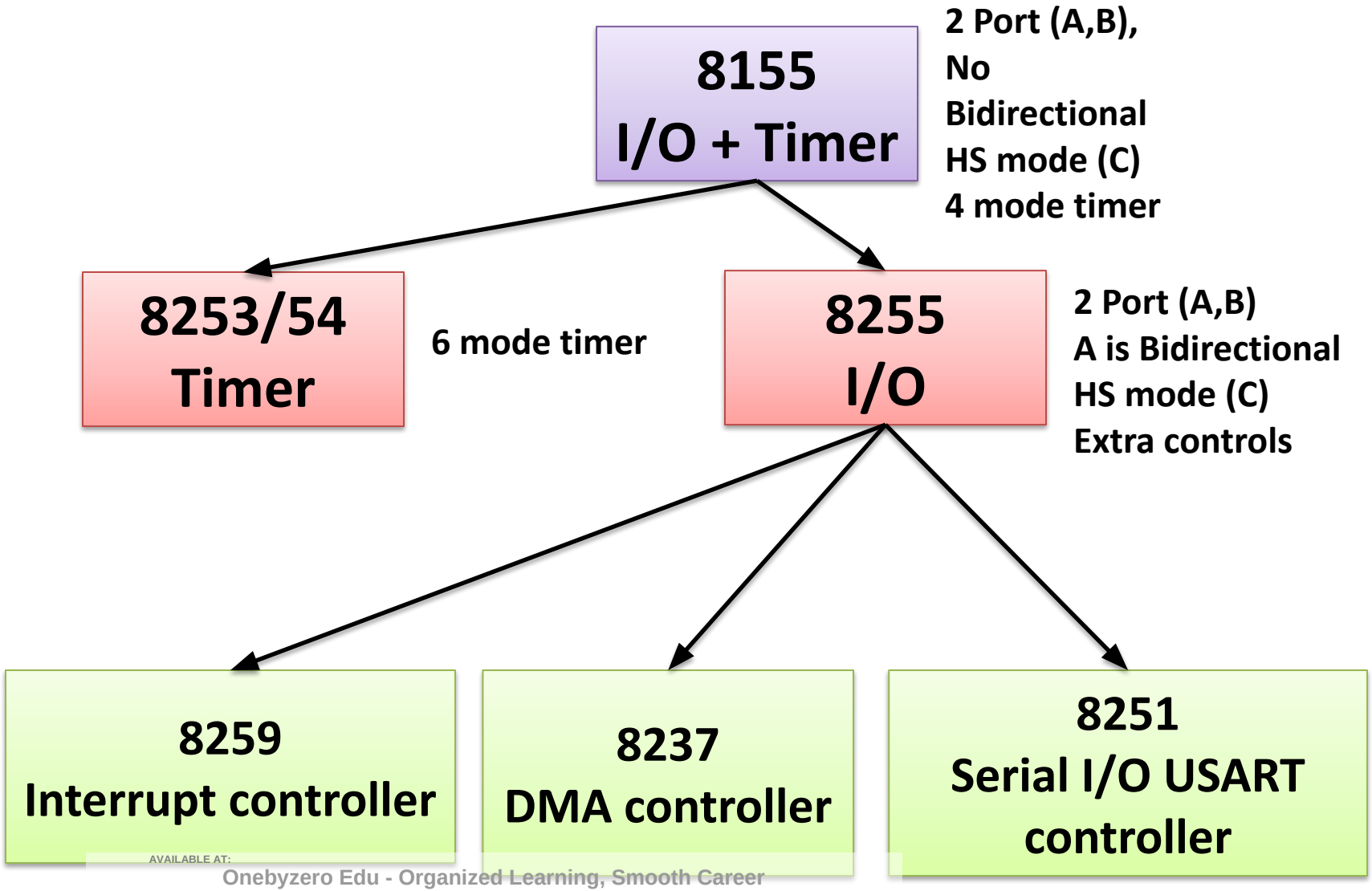
Peripheral Interface Device (8255 modes and examples)

Dr A Sahu
Dept of Computer Science & Engineering
IIT Guwahati

Hierarchy of I/O Control Devices

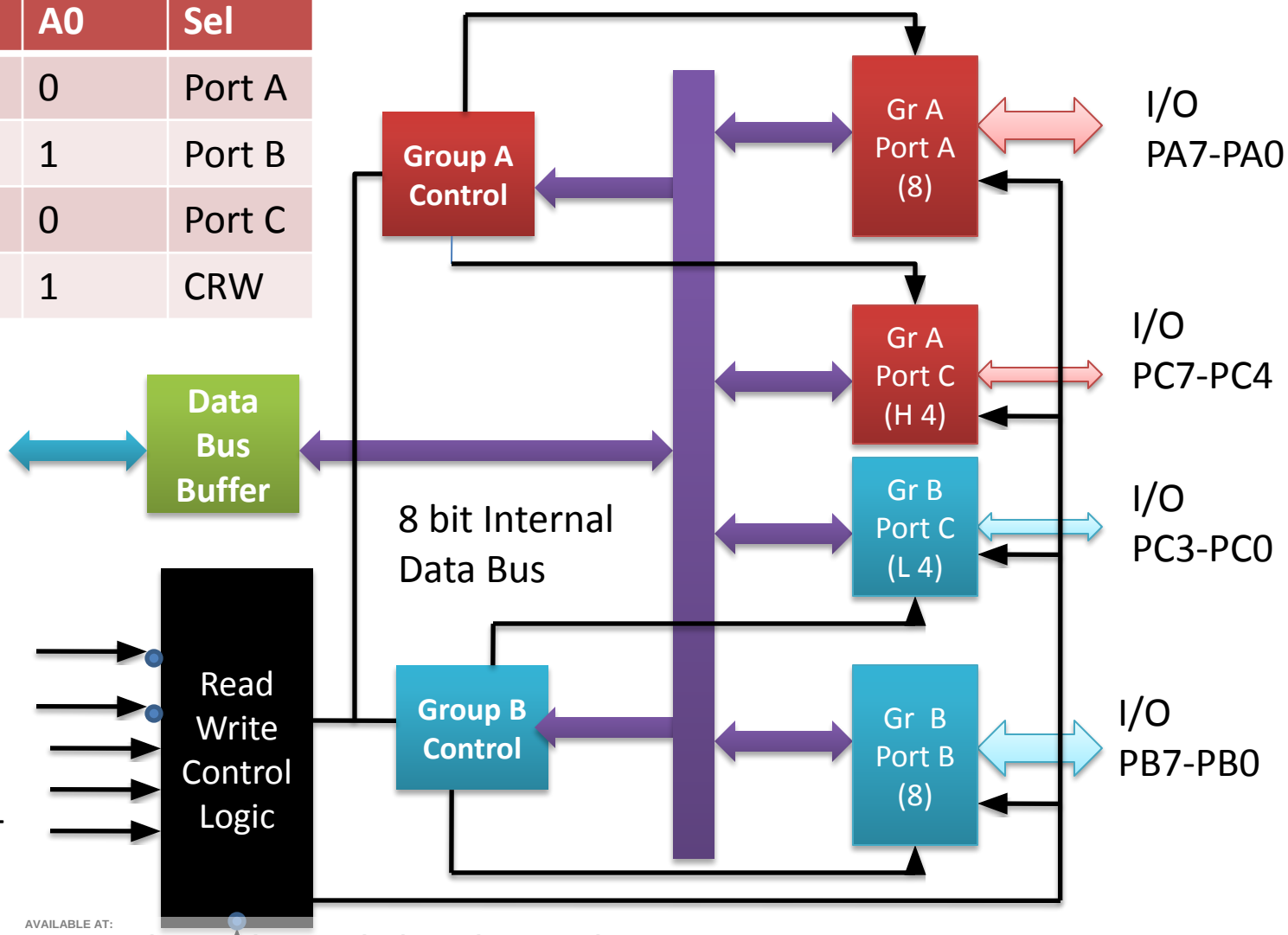


Outline

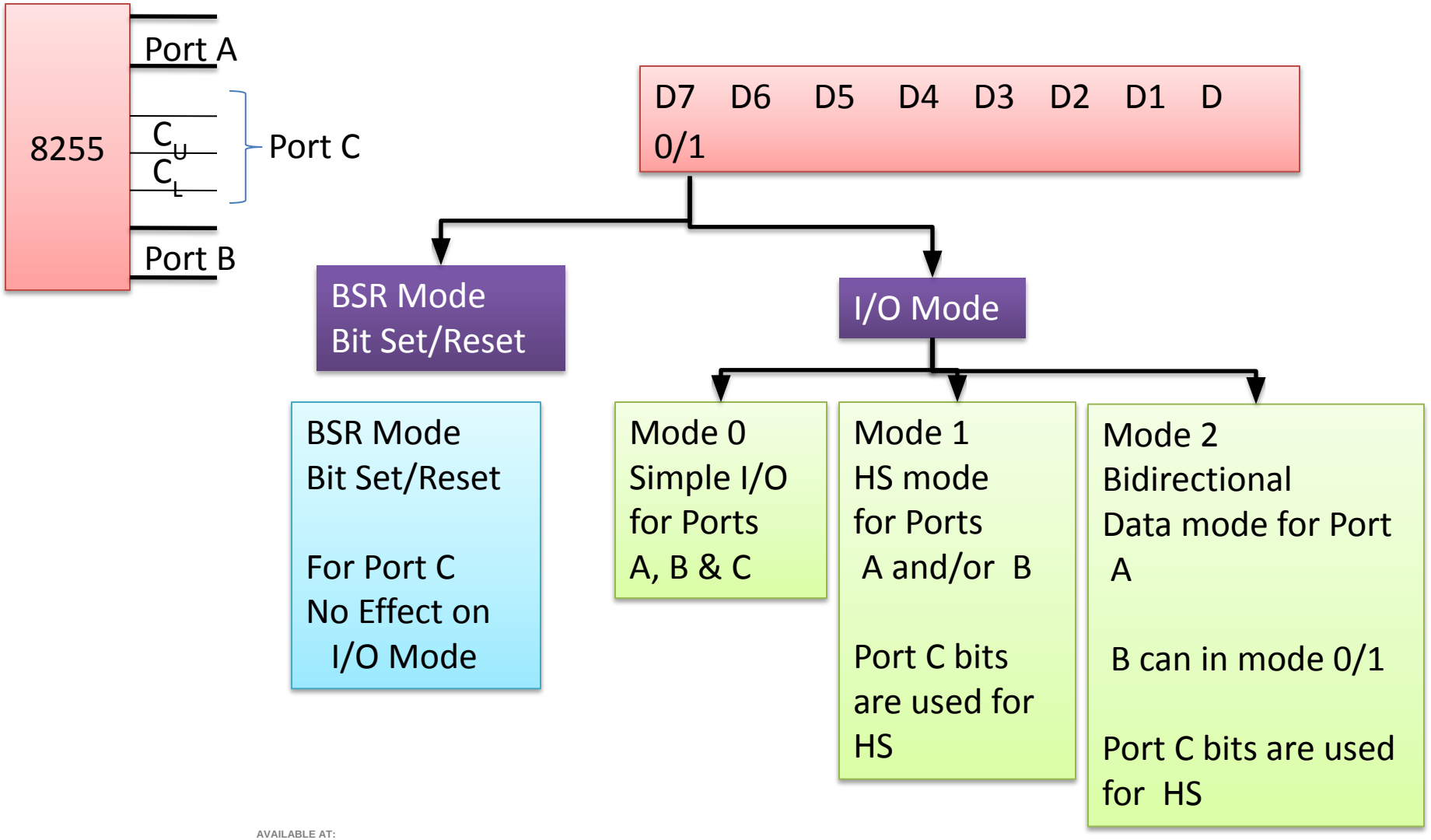
- 8255 Architecture and its block diagram
- 8255 Ports and mode of operations
- BSR Mode
- Mode 0
  - Interfacing A/D Converter using Handshake mode using 8255
- Mode 1
  - Interfacing DIP keyboard using Handshake mode using 8255
- Mode 2
  - Interfacing keyboard (with bounce) using Handshake mode using 8255
- Introduction to interrupt controller (8259A)

Block Diagram of 8255

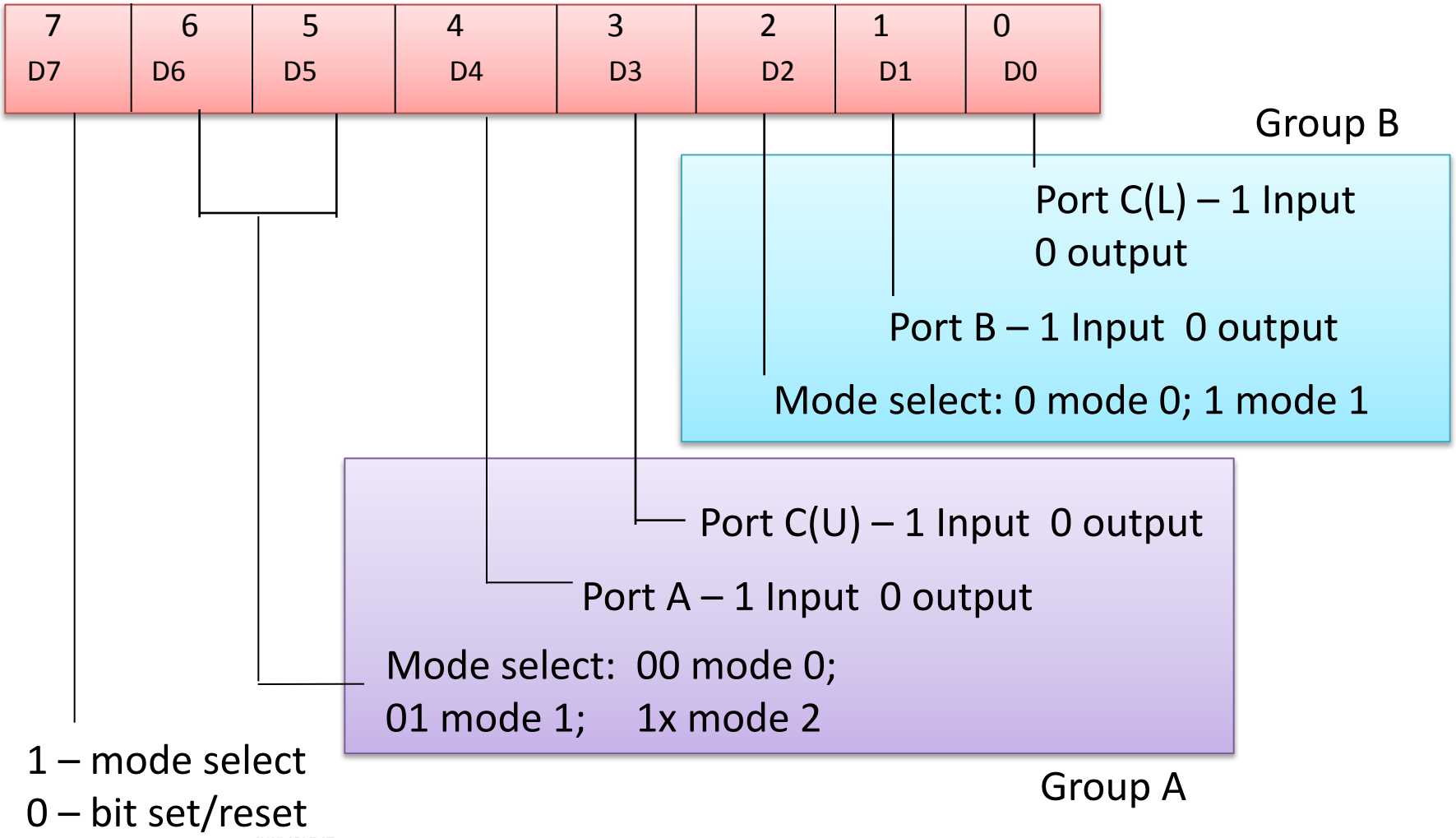
CS^b	A1	A0	Sel
0	0	0	Port A
0	0	1	Port B
0	1	0	Port C
0	1	1	CRW



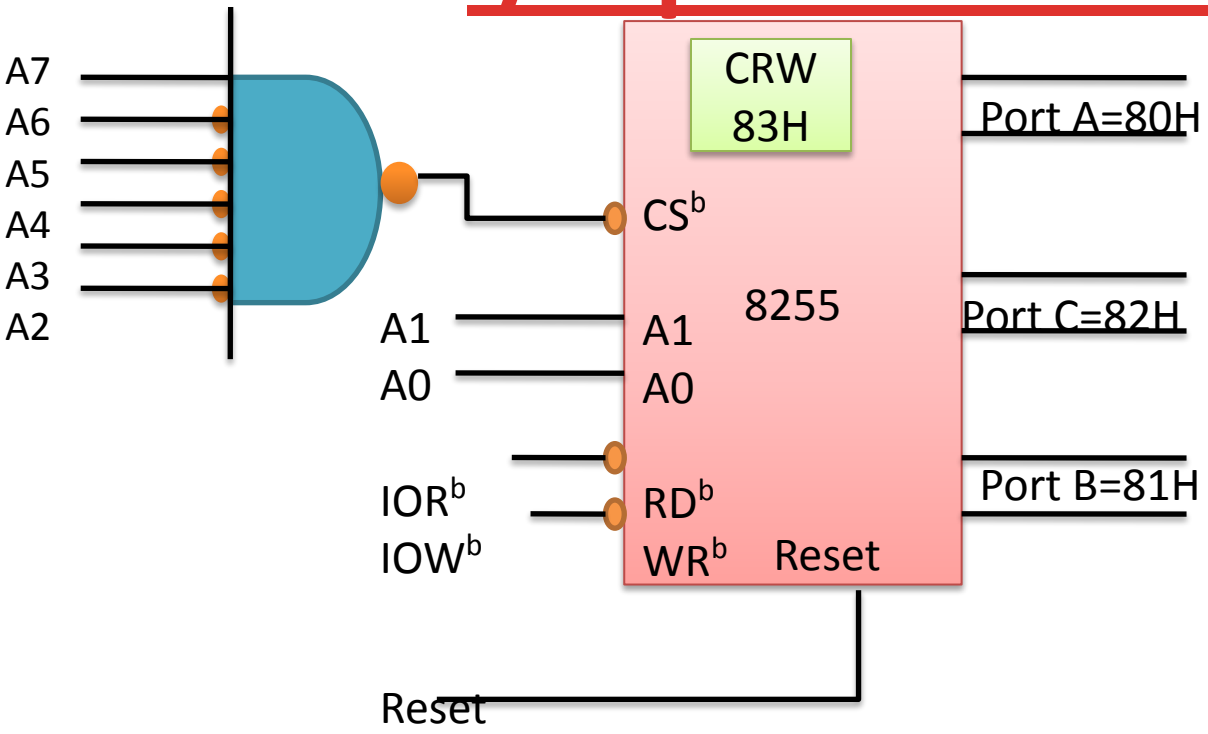
Ports & Modes in 8255



Ports & Modes in 8255 : Control register



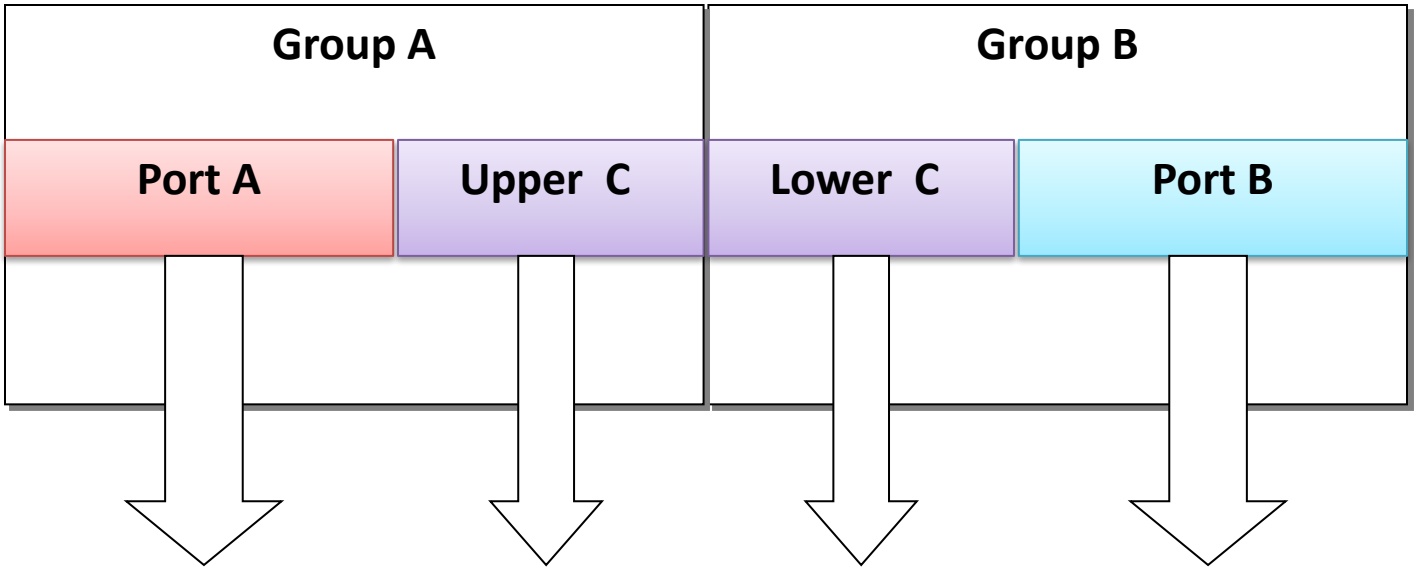
I/O port Addressing



CS^b	A1 A0	HEX Address	Port
A7 A6 A5 A4 A3 A2 1 0 0 0 0 0	A1 A0 0 0	= 80H	A
	0 1	=81H	B
	1 0	=82H	C
	1 1	=83H	Control Register

Ports

- Control register controls the overall operation of 8255
- All three ports A, B and C are grouped into two



Operation modes

- 8255 has three modes:
  - Mode 0: basic input-output
  - Mode 1: Strobbed input-output
  - Mode 2: Strobbed bi-directinal bus I/O
- In mode 0
  - Two 8-bit ports and two 4-bit ports
  - Any port can be input or output
  - Outputs are latched, inputs are not latched

Operation modes

- In mode 1:

  -Three ports are divided into two groups

  -Each group contains one 8-bit port and one 4-bit control/data port

  - 8-bit port can be either input or output and both latched

  - 4-bit port used for control and status of 8-bit data port

- In mode 2

  - Only port A is used

  - Port A becomes an 8-bit bidiectional bus

  - Port C acts as control port (only pins PC3-PC7 are used)

BSR (Bit Set or Reset Mode)

- Set/Reset bit of Port C
- Heavily used for HS and Interrupt mode
- BSR Control word

D7	D6	D5	D4	D3	D2	D1	D0
0 BSR Mode	Not used, So (000)			Bit Select			S/R (1/0)

- BSR Control word
  - To set PC7= 0 000 111 1 (0FH)
  - To reset PC7= 0 000 111 0 (0EH)
  - To set PC3 = 0 000 011 1 (07H)

BSR Mode example

- BSR Control word
  - To set PC7= 0 000 111 1 (0FH)
  - To reset PC7= 0 000 111 0 (0EH)
  - To set PC3 = 0 000 011 1 (07H)

D7	D6	D5	D4	D3	D2	D1	D0
0 BSR Mode	Not used, So (000)			Bit Select			S/R (1/0)

Generate Activation pulse of Delay D on PC7&PC3

```
MVI     A,0FH          ; Load ACC to set PC7
OUT     83H       ; set PC7=1
MVI     A,07H          ; Load ACC to set PC3
OUT     83H       ; set PC3=1
CALL    DELAYD;
MVI     A,06H          ; Load ACC to Reset PC3
OUT     83H       ; set PC3=1
MVI     A,0EH          ; Load ACC to Reset PC7
OUT     83H       ; set PC7=1
```

8255: Mode 0

- Simple I/O for port A,B,C
- Output are latched
- Input are not latched
- Port don’t have HS or interrupt capability

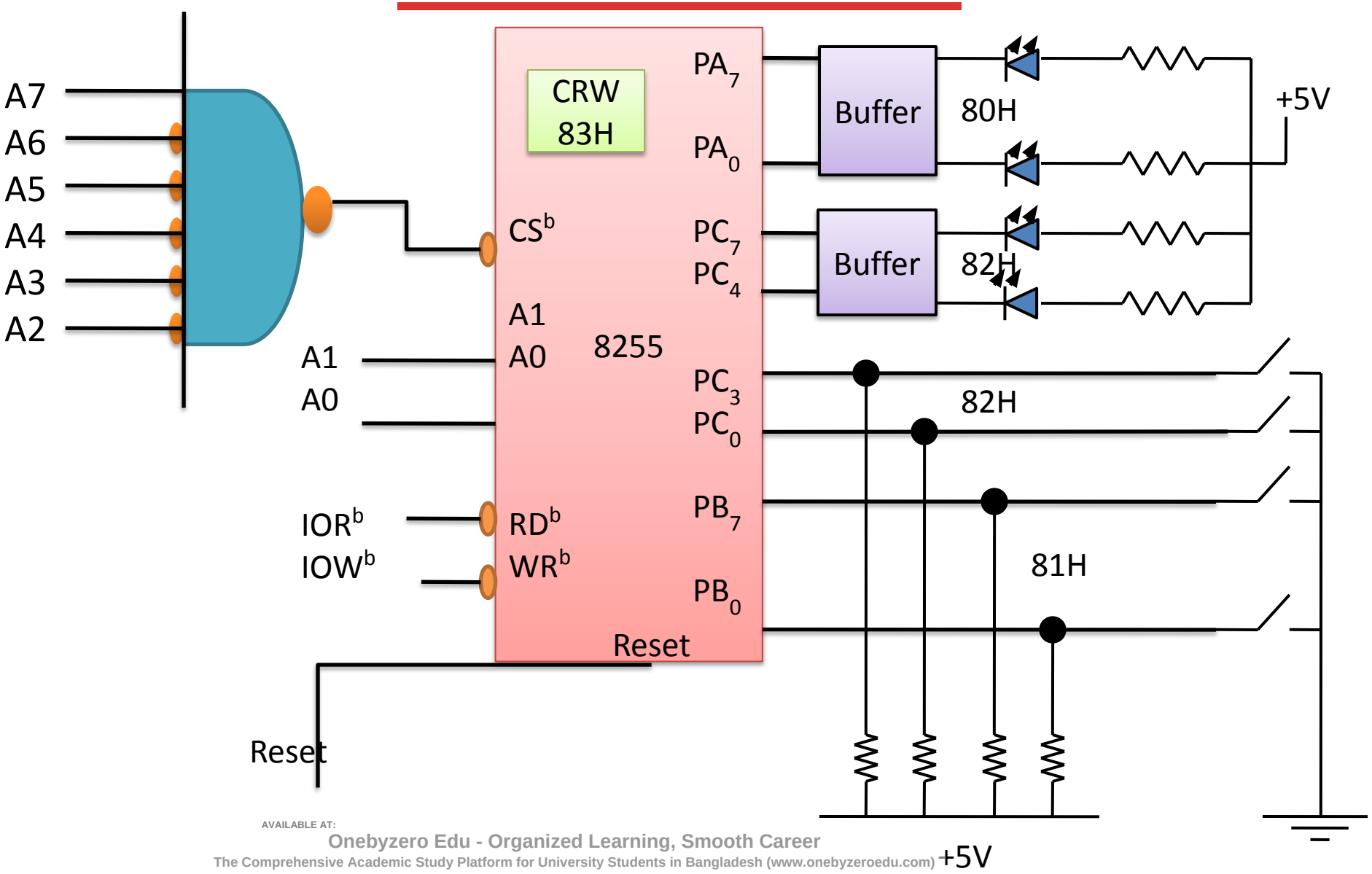
8255: Mode 0, Example 1

- Configure
  - Port A and port C_U as out port
  - Port B and port C_L as in port
- Interface to Read from I/P DIPs and Display at O/P LEDs
- Control word

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	1	1
I/O function	Port A in Mode 0		Port A as O/P	Port C_U As O/P	Port B in Mode 0	Port B As I/P	Port C_L As I/P

83H

Interface Circuit



Interface Program

```
MVI   A,83H     ; Load acc  with Control word
OUT   83H       ; Load control register with 83
    at port address 83
IN81H           ; Read DIP from port B
OUT   80H       ; Write to LEDs
IN 82H          ; Read DIP from port  C
ANI   0FH       ; Mask upper part of
                port C are not i/p
RLC RLC  RLC RLC; Rotate 4 time
OUT   82H       ; Display data at port C
HLT
```

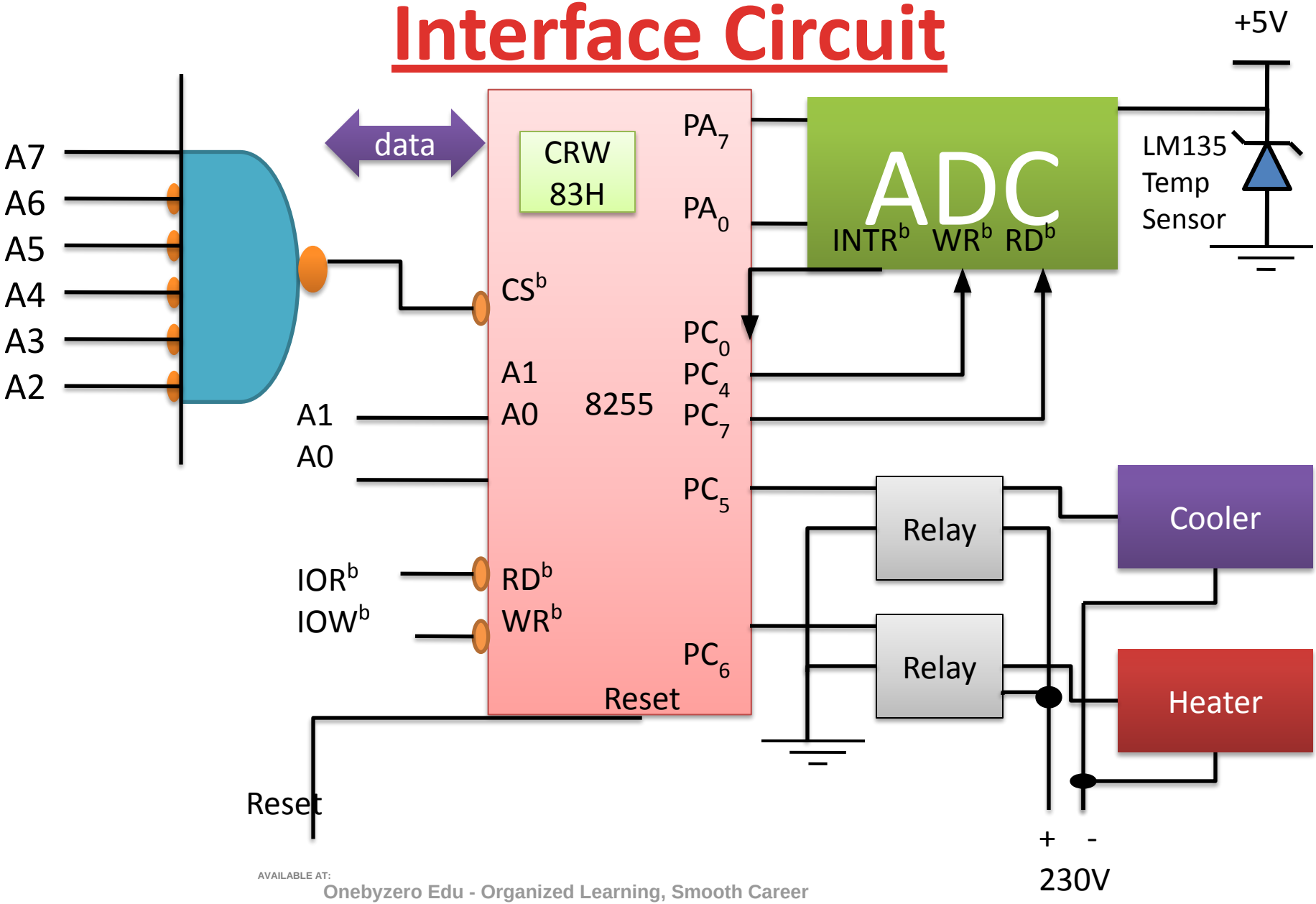
8255: Mode 0, Example 2

- Air Conditioning Room (Temperature Control)
  - Heater and Cooler
  - Temperature Sensor
  - A/D converter
  - Driver Switch to drive Heater/Cooler
- Design an A/C controller using 8255 and 8085 based interfacing circuit
- Read temperature and control the temperature between 20-30 degree Celciuous

8255: Mode 0, Example 2

- Air Conditioning Room (Temperature Control)
  - Heater and Cooler
  - Temperature Sensor
  - A/D converter
  - Driver Switch to drive Heater/Cooler
- Design an A/C controller using 8255 and 8085 based interfacing circuit
- Read temperature and control the temperature between 20-30 degree Celsius
- **Use port A in mode 0 and Port C in BSR mode**

Interface Circuit



Interface control

- Control word
  - Port A as I/P from ADC
  - Port C_L : as I/P PC_0 is used for end of conversion
  - Port C_U : as O/P PC_4 -> Start con. PC_7 ->assert RD^b signal

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	1	0	0	0	1
I/O function	Port A in Mode 0		Port A as I/P	Port C_U As O/P	Port B Is Not used		Port C_L As I/P

91H

- BSR Control word
  - 0 (mode) 000 (don't care) 000 (0/1=set/reset)
  - Set PC7 high = 0 000 111 1 = 0FH (Send RD^b to ADC)
  - Set PC4 low = 0 000 100 0 = 08F (send Start Conv WR^b)
  - Set PC5 high = 0 000 101 1 = 0BF (Fan On)
  - Set PC5 low = 0 000 101 0 = 0AF (Fan Off)

Interface Program to do Temp. Control

```
      MVI  A, 91H  ; mode 0 control word
      OUT  83H     ; Set A& CL as I/P & CU as
                     O/P
      MVI  A,0FH   ;Set PC7 High
      OUT  83H     ; Disable RDb
      MVI   A,08H  ; Set PC4 WRb low
      OUT   83H    ;  Start conversion
      MVI   A, 09H ; Set PC4 WRb high
      OUT  83H     ; Ser WRb high

RD:   IN   82H     ; Read Port C to Chck PC0
      RAR          ; Place PC0 in Carry Flag
      JC  RD       ;if PC0=1, read Again
      MVI A,0EH    ; Set PC7 RDb low
      OUT  83H     ; Assert RDb signal
      IN 80H       ; Read A/D conv. Port A
      MOV  B,A     ; get temp in B
```

```
      MVI  A   0FH; ; Set PC7 (RDb) high
      OUT 83 ; Disable RDb
      MOV A,B;
      CPI 30D;
      CNC   COOLEROFF;      PC5 off  0AH
      CC    COOLERON;       PC5 on 0BH
      CPI  10D;
      MOV A,B;
      CNC   HEATERON;       PC6on:  0CH
      CC    HEATEROFF;      PC6off: 0DH
      RET

COOLEROFF:
      MVI  A, 0AH ; Reset PC5 to turn off Cooler
      OUT 83H
      RET
```

8255: Mode 1

- Two port A & V function 8 bit I/O
  - Configured either Input or output port
- Each port each 3 lines of port C as HS signal
  - Remaining two lines can be used as simple I/O
- Input and output are latched
- Interrupt logic is supported

PC6, PC7 in In/Out mode

D7	D6	D5	D4	D3	D2	D1	D0
1	**0**	**1**	0	0/1	**1**	1	X=0
I/O function	**Port A in Mode 1**		Port A as O/P	Port C_U As O/P	**Port B in Mode 1**	Port B As I/P	Port C_L As I/P

AEH

8255: Mode 1: Input Control signal

- STB^b: Strobe generated by Peripheral
- IBF: Input buffer full
  - Acknowledge by 8255 to I/O that I/O latched received
  - Reset when MPU read the data
- INTR: output signal to MPU and it generate when STB^b=1, IBF=1 and INTE=1
- INTE: Enable or disable Interrupt
  - $INTE_A$ is through PC_4, $INTE_B$ is through PC_2
- Status word

D7	D6	D5	D4	D3	D2	D1	D0
OBF^b_A	$INTE_A$	1/0	1/0	$INTR_A$	$INTE_B$	OBF^b_B	$INTR_B$

Mask with 02H

8255: Mode 1, Example

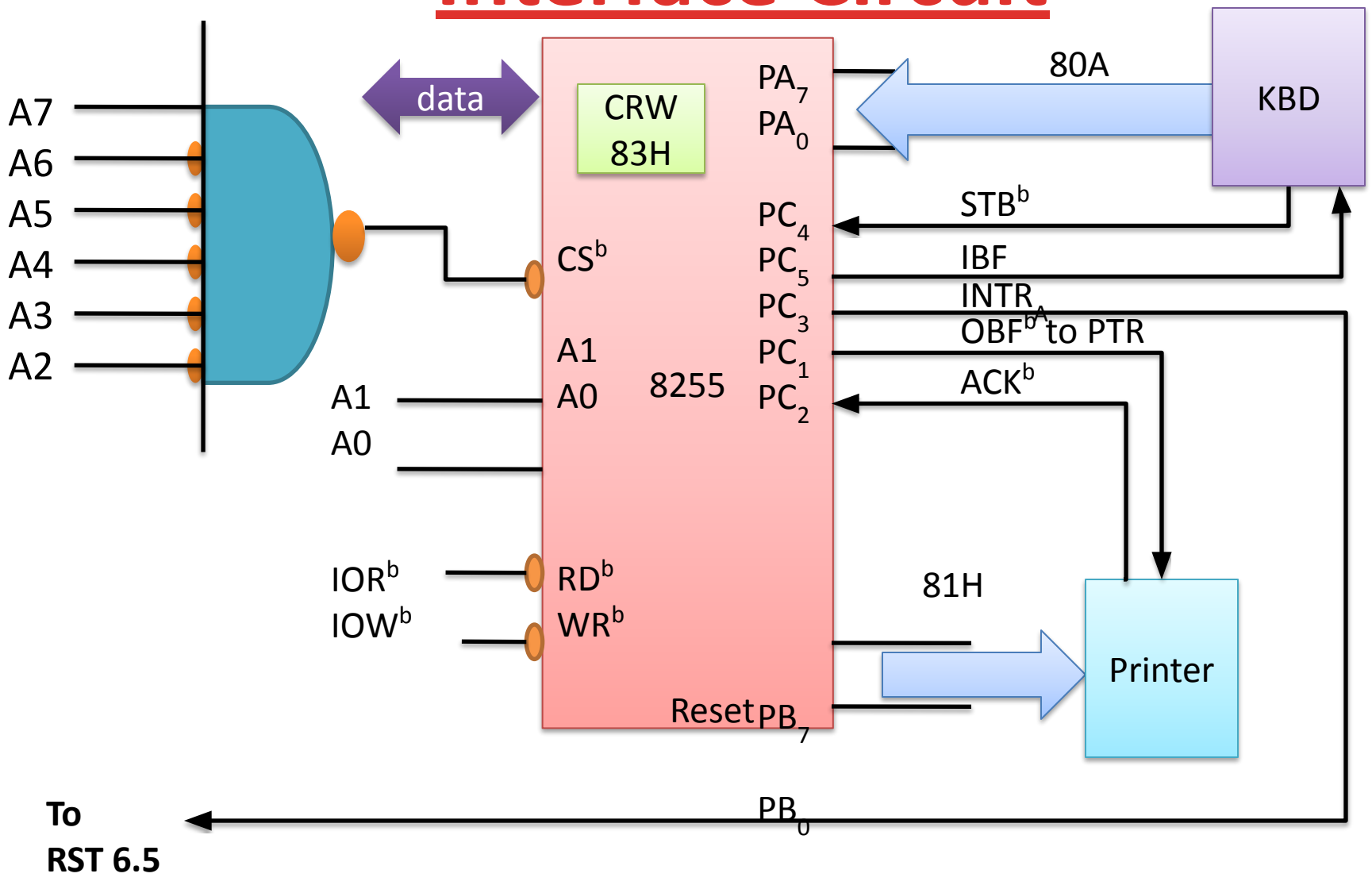
- Designing for interfacing
  - Keyboard with interrupt I/O in port A
  - Output port for a printer using status check I/O
- Control word

D7	D6	D5	D4	D3	D2	D1	D0
1	0	1	1	0	1	0	0
I/O function	Port A in Mode 1		Port A as I/P	$PC_{6,7}$ as X	Port B in Mode 1	Port B As O/P	Don't care

B4H

- To generate interrupt $INTE_A$ PC_4 to set in BSR mode
  - 0 (mode) 000 (don't care) 000 (0/1=set/reset)
  - Set PC4 High = 0 000 100 1 = 09F (send INTR to MPU RST 6.5)

Interface Circuit



Interface Program

Initialization Program

```
MVI  A, B4H ; initialize port A as IP and B
              as O/P
OUT 83H
MVI A,09H  ; Set INTEA , that is PC4
EI              ;Enable interrupt
CALL PRINT   ; Continue other Task
```

ISR at 0034 at RST6.5 vector location

0034: JMP READPORTA

```
READ PORTA:
                DI
          IN 80H
          MOV M, A
          INX H
          EI
          RET
```

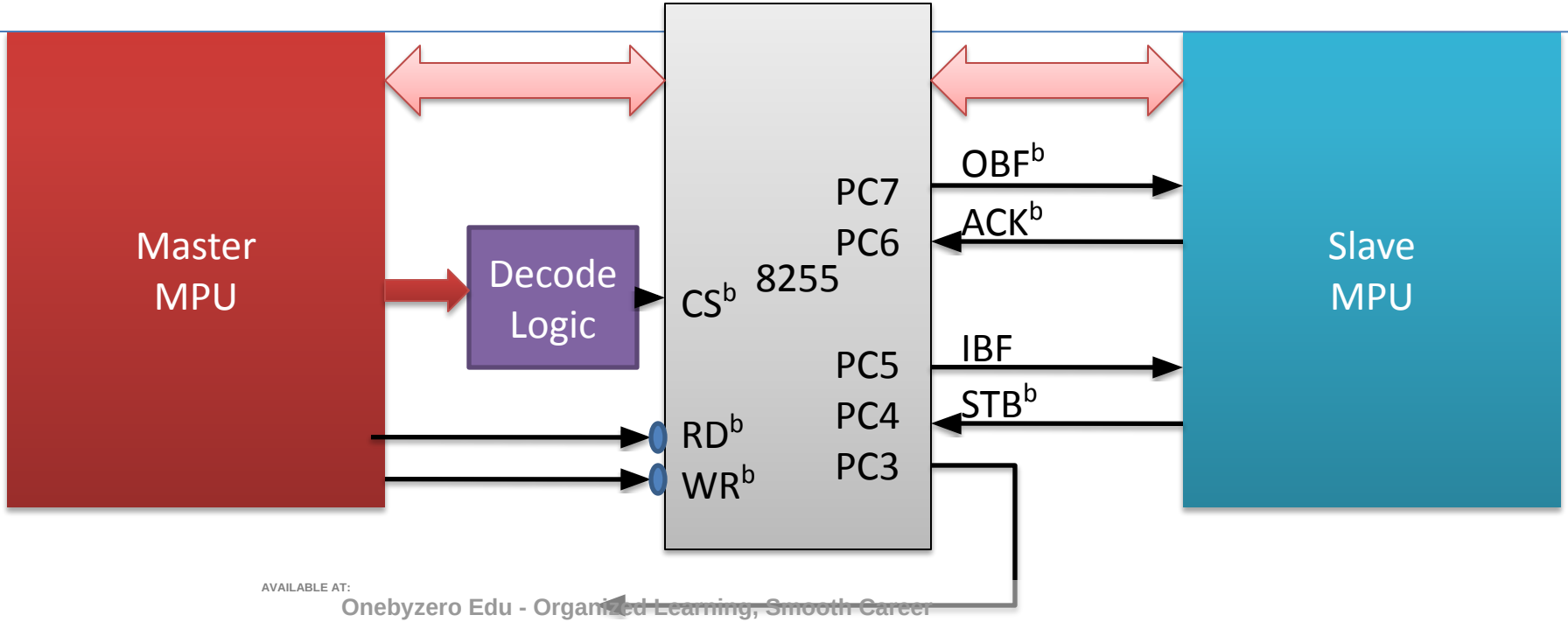
```
PRINT:    LXI H MEM
      MVI  B COUNT
      MOV A,M
      MOV C,A
STATUS:   IN   82H ; from port C for
                     Status OBF^b
          ANI 02H
      JNZ STATUS

          MOV A,C
      OUT 81H; Send to port B
                   printer
      INX H
      DCR B
      JNZ NEXT
      RET
```

8255: Mode 2: Bi-directional Data transfer

- Bi-directional
  - Data transfer between two MPU
  - Data transfer between MPU and Controller
- **Port A can be bi-directional**, Port B in either 0 or 1 mode
- **Port A use 5 signals from port C as Handshake signal for data transfer**

Bi-directional Data transfer between two MPU

- One is Master other is Slave
- Use 8255 as Interfacing device



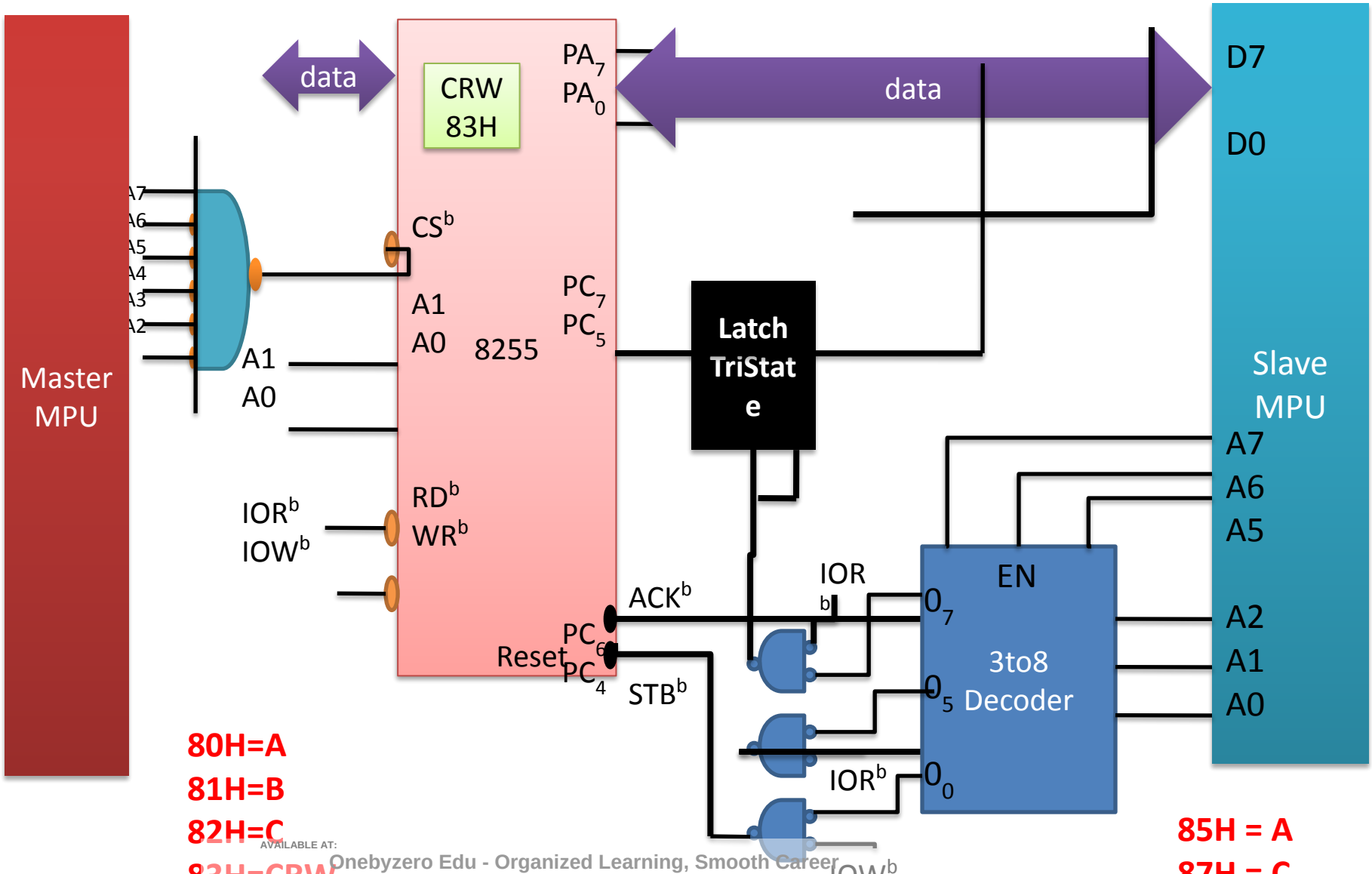
Data Transfer From Mater to Slave

- Master read the status of OBF to verify whether the previous byte has read by Slave
- Mater **write date in port A** and 8255 inform to Slave by OBF^b low
- Slave check OBF for data availability
- Slave **read the data from port A** and ACK low to 8255

Data Transfer From Slave to master

- Slave check the HS signal IBF to find out whether port A is available or not
- Slave **Write a data in port A** and inform 8255 by enabling STB^b low
- 8255 causes a IBF to go high and MPU get the signal the data byte to read
- Master **read the data from port A** and make IBF low

Interface Circuit



Control word Mode 2

- Control word

D7	D6	D5	D4	D3	D2	D1	D0
1	1	X	X	X	1/0	1/0	1/0
I/O function	Port A in Mode 1		Port A as Bi		Port B in Mode 1/0	Port B As 1/0	Port C

C0H

Port C bit 2,1,0 mode 0/1

- Status word:

D7	D6	D5	D4	D3	D2	D1	D0
OBF_A	$INTE_1$	IBF_A	$INTE_2$	$INTR_A$	X	X	X

RAL instruction to get the Status

Interface program: Master & Slaves

Master:

```
        LXI  H, MemptrM
        MVI B, Byte2Trasfer
        MVI  A, CTRL;   Control word for
                        Mode 2
        OUT  83H;    Write Control word

OBFLO:
        IN 82H ; Read port C
        RAL    ; place OBF in CY
        JNC OBFLO;

        OUT  80H ; place on Port A
        INX H
        DCR B
        JNZ OBFLO

        HLT
```

SLAVE:

```
        LXI  H, MemptrS
        MVI B, Byte2Trasfer

OBFHI:
        IN 87H ; Read port C
        RAL    ; place OBF in CY
        JC OBFHI;

        IN  85H ; Read from  Port A
        MOV M, A

        INX H
        DCR B
        JNZ OBFHI
        HLT
```

Introduction to Interrupt controller 8259A

- Acts as a multiplexer, combining multiple interrupt input sources into a single interrupt output to interrupt a single device.
- Original PC introduced in 1981
- Eight interrupt input request lines
  - IRQ0 - IRQ7,
  - An interrupt request output line named INTR
  - Interrupt acknowledgment line named INTA
  - D0 through D7 for communicating the interrupt level or vector offset.
- There are three registers
  - Interrupt Mask Register (IMR)
  - Interrupt Request Register (IRR)
  - In-Service Register (ISR)

Reference

- R S Gaonkar, “Microprocessor Architecture”, Chapter 15

Thanks