

Department of Electrical Engineering

University of Arkansas



ELEG3923 Microprocessor Analog to Digital Converter (ADC)

Dr. Jingxian Wu

wuj@uark.edu

ADC

- **Analog to digital converter (ADC)**

- Convert analog signal to digital signal

- Analog signal: a signal that has continuous values

- E.g. temperature

- Digital signal: a signal that can only take discrete values

- E.g. 0 or 1

- E.g. 0, 1, 2, 3

- ADC: convert analog signal to digital signal

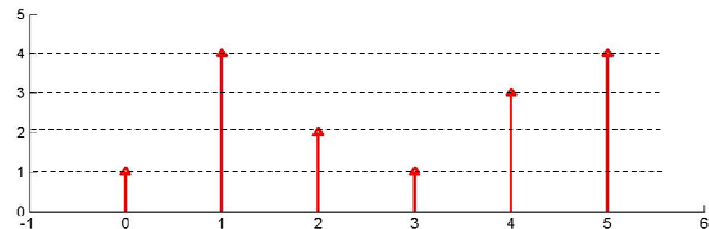
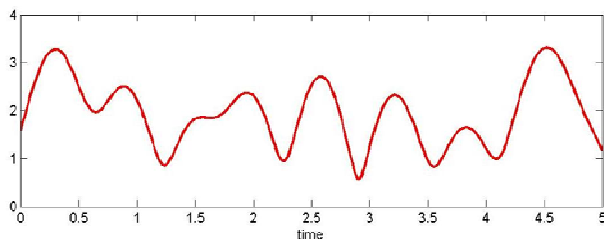
- Example

- 0 ~ 5 V is divided into 4 levels (each level can be represented by 2 bits)

- Step size: $5/4 = 1.25$ V

- Analog: 0 ~ 1.25V → digital: 00; analog: 1.25 ~ 2.5V → digital: 01

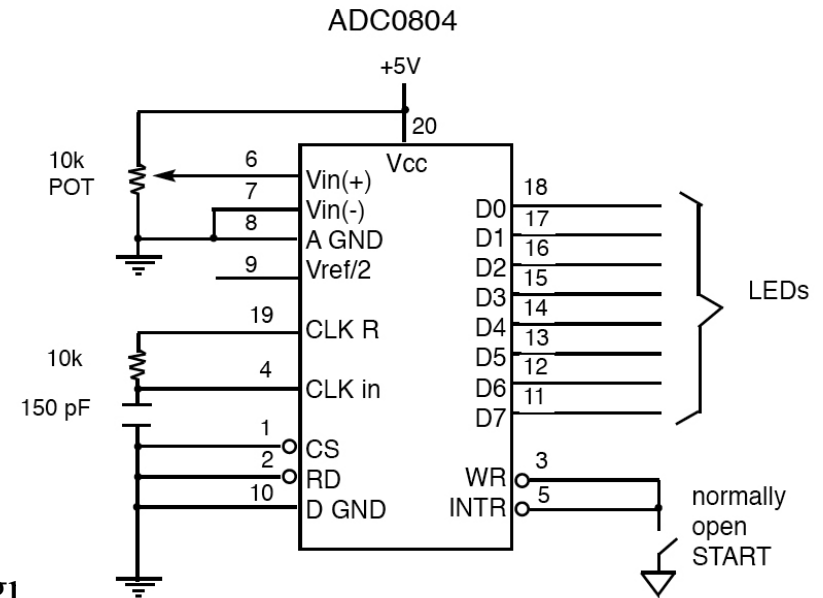
- Analog: 2.5 ~ 3.75V → digital: 10; analog: 3.75 ~ 5V → digital: 11



ADC

- **ADC0804**

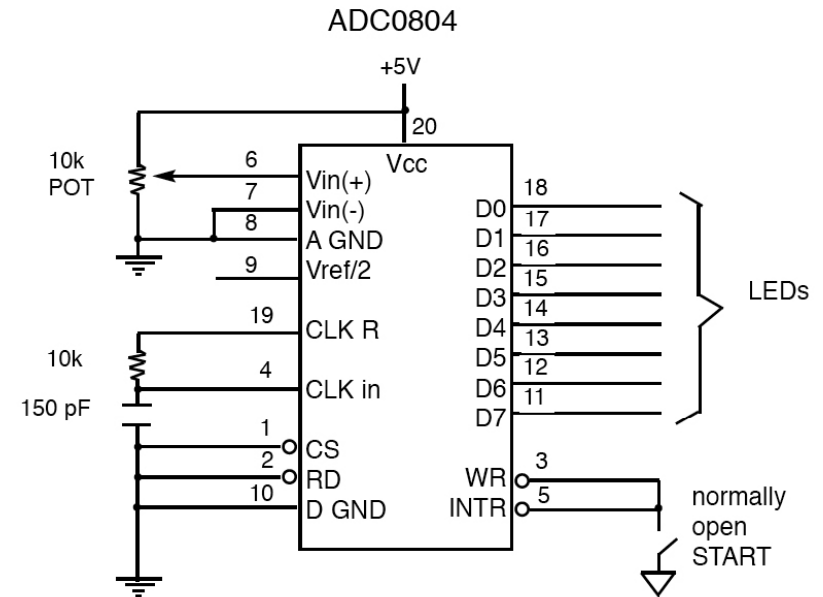
- 8-bit parallel ADC, 0 ~ 5V
- Each analog level is represented by 8-bit
 - 0 ~ 5V is divided into $2^8 = 256$ levels.
- Control pins
 - CS (chip select)
 - CS = 0: activate ADC0804
 - RD (read)
 - RD = 0: read the converted digital signal
 - WR (write, or start conversion)
 - When CS = 0 and WR L-to-H, then ADC0804 will convert the analog signal $X = V_{in(+)} - V_{in(-)}$ to digital value and store in its internal register
- Input pins
 - $V_{in(+)}$, $V_{in(-)}$, the voltage difference between the two pins is the input analog signal to be converted



ADC

- **ADC0804**

- CLR in, CLR R
 - Used to control the conversion time
 - $T = 1.1RC$
 - It takes to duration T to finish one conversion
- INTR (end of conversion)
 - When conversion is done, INTR becomes 0
- Data pins
 - D0 ~ D7: the 8 bits digital data (256 discrete levels)
 - Step size: $p = 5V/256$
 - E.g $D = 00H \rightarrow V_{in} = 0V$, $D = 01H \rightarrow V_{in} = pV$,
 - $D = FFH \rightarrow V_{in} = 5V$
 - $V_{in} = D * p$



ADC

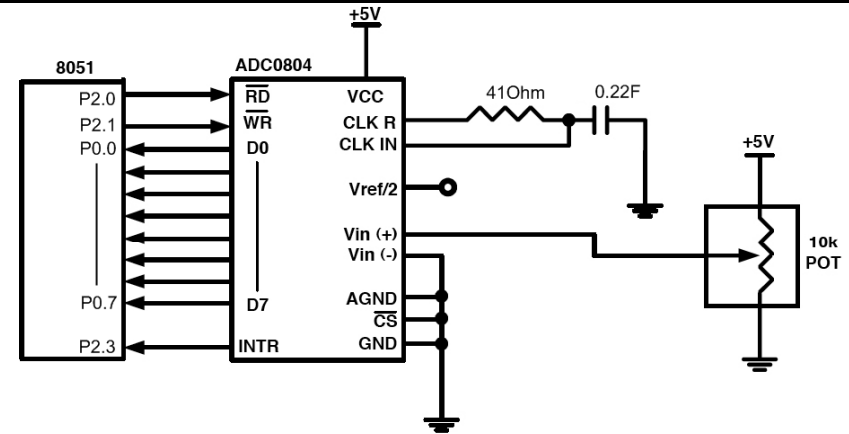
- **Connect ADC0804 to 8051**
- **Sample program**

```
RD BIT P2.0
WR BIT P2.1
INTR BIT P2.3
MYDATA EQU P0
```

```
MOV P1, #0FFH      ; input port
SETB INTR          ; input pin, monitor if conversion is over

BACK: CLR WR
SETB WR           ; L-to-H to start conversion

HERE: JB INTR, HERE ; wait for end of conversion
CLR RD           ; conversion done, enable RD to read data
MOV A, MYDATA    ; read data into A
ACALL BIN2ASCII
ACALL DATA_DISPLAY
SETB RD          ; make RD = 1 for next round
SJMP BACK
```



TEMPERATURE SENSOR

- **Temperature sensor**

- Sense the temperature, convert the temperature into voltage
- LM34
 - $V_S = 5V$
 - V_{out} : the voltage that is proportional to temperature
 - 10 mV/°F
 - $V_{out} = 0$: 5°F, $V_{out} = 10mV$: 6°F, $V_{out} = 20mV$: 7°F
 - $T = V_{out}/10mV + 5$

