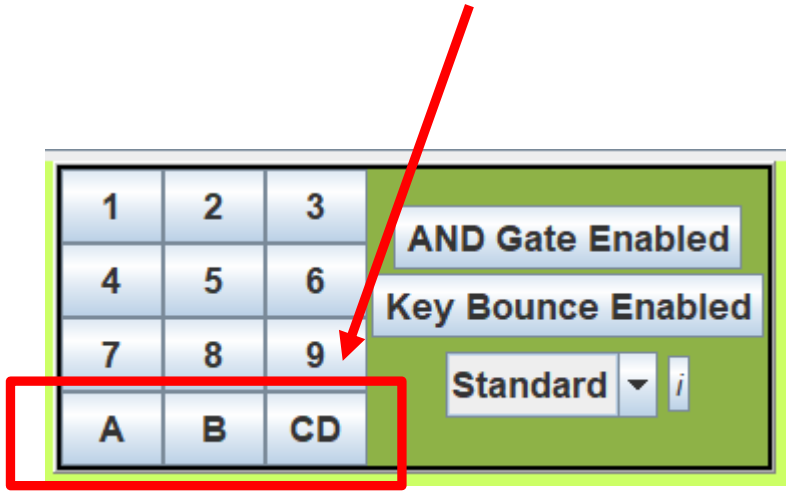
Arquitetura de Computadores

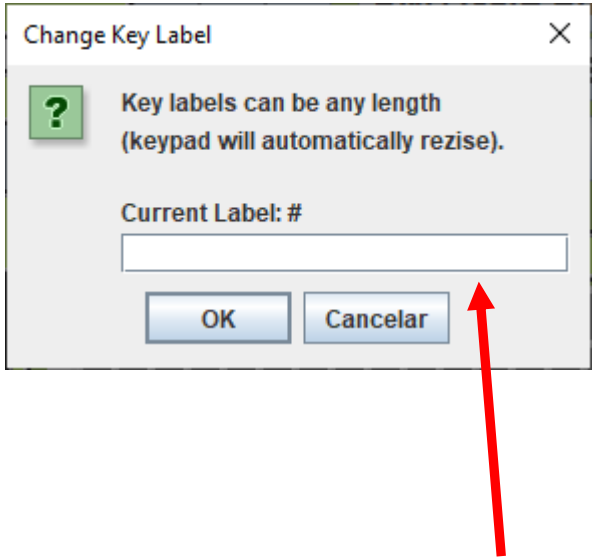
PROF. ISAAC

Personalizando o EdSim51 para Projeto

Modificando as Teclas do Teclado Matricial

No edSim51 você pode modificar as teclas do seu teclado matricial, para configurá-lo de acordo com o seu projeto.

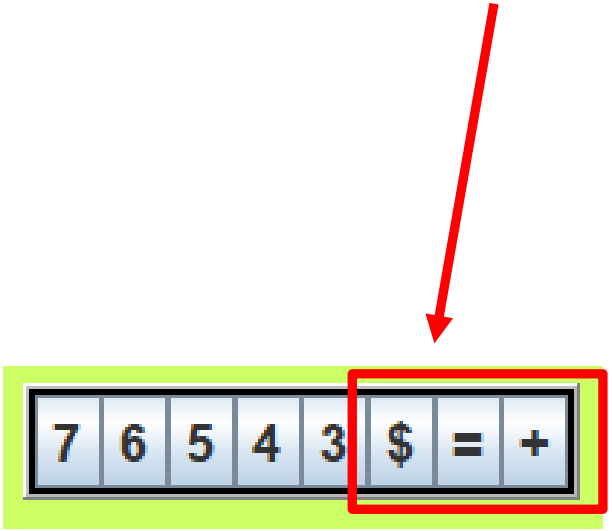


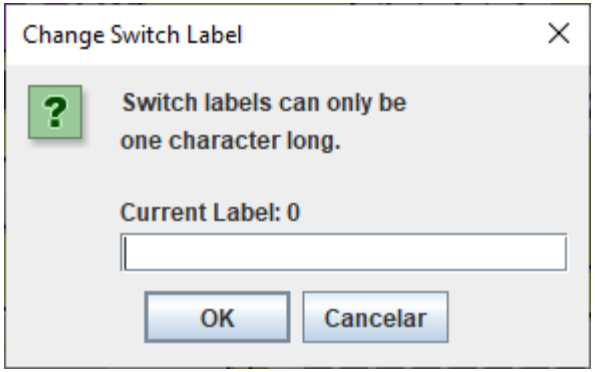


Para modificar, basta clicar com o botão direito sobre a tecla e escrever o novo *Label* da tecla.

Modificando os *label* dos botões

No edSim51 você pode modificar os *label* dos botões para configurá-lo de acordo com o seu projeto.





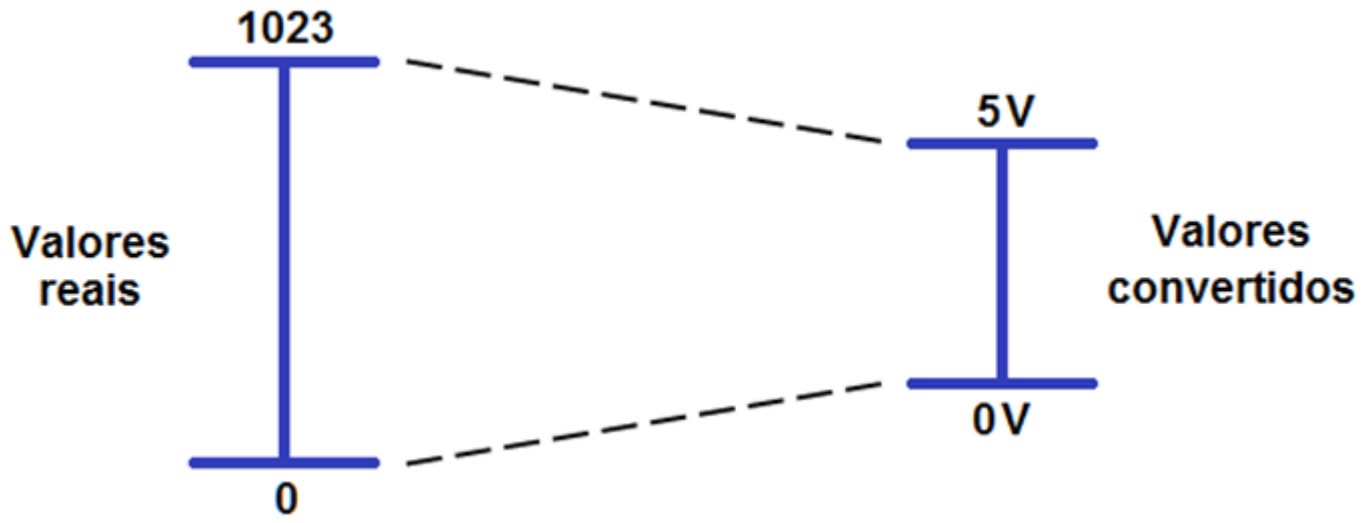
Para modificar, basta clicar com o botão direito sobre o botão e escrever o novo *Label*.

ADC e DAC

DAC

DAC - Digital to Analog Converter

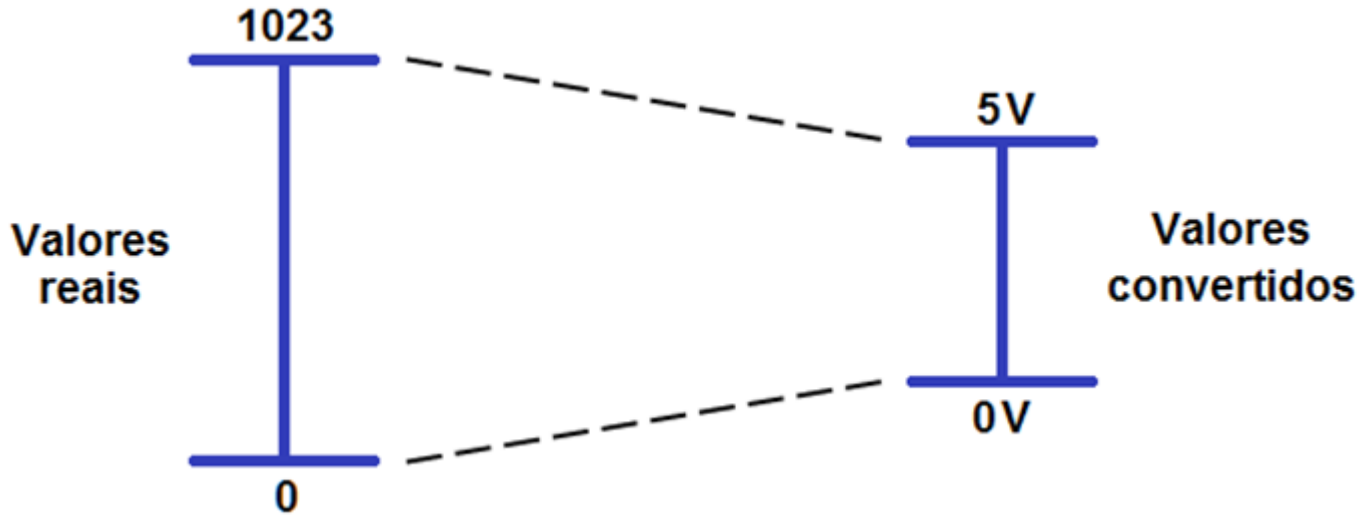
O conversor digital-analógico (DAC), é um circuito eletrônico que converte uma grandeza digital (código binário) em uma grandeza analógica (normalmente uma tensão ou uma corrente).



DAC - Digital to Analog Converter

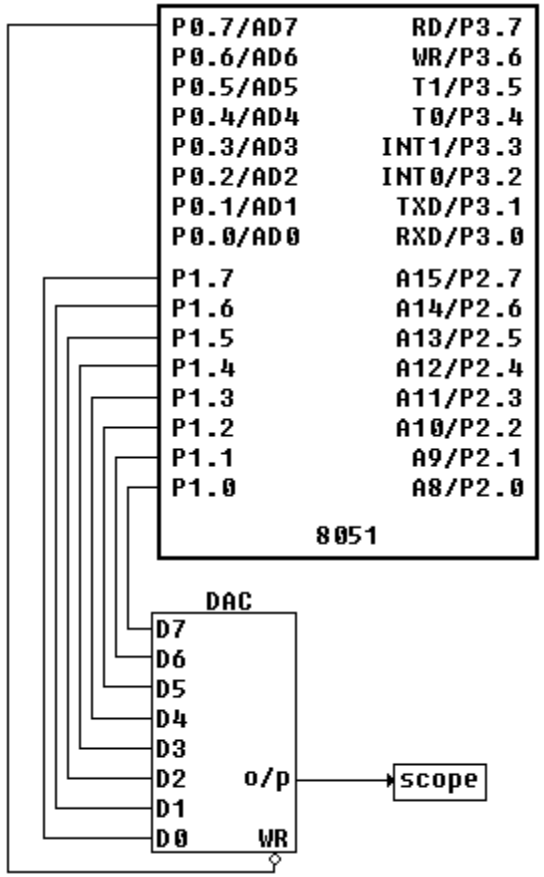
Exemplo:

Podemos converter um sinal digital de números de 0 a 1023 para um sinal analógico de tensão de 0 a 5V.



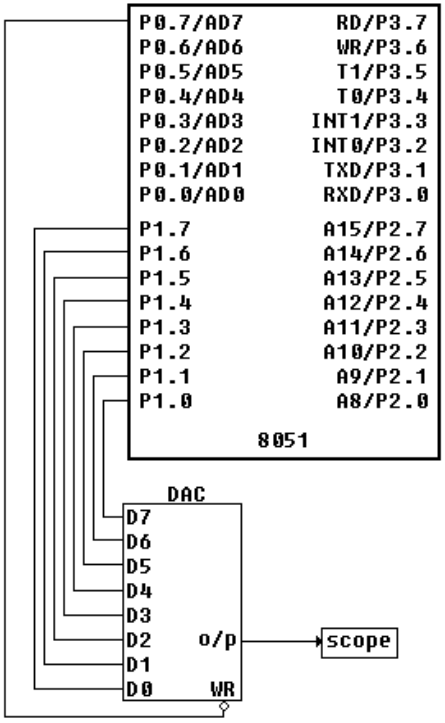
DAC - Digital to Analog Converter – EdSim51

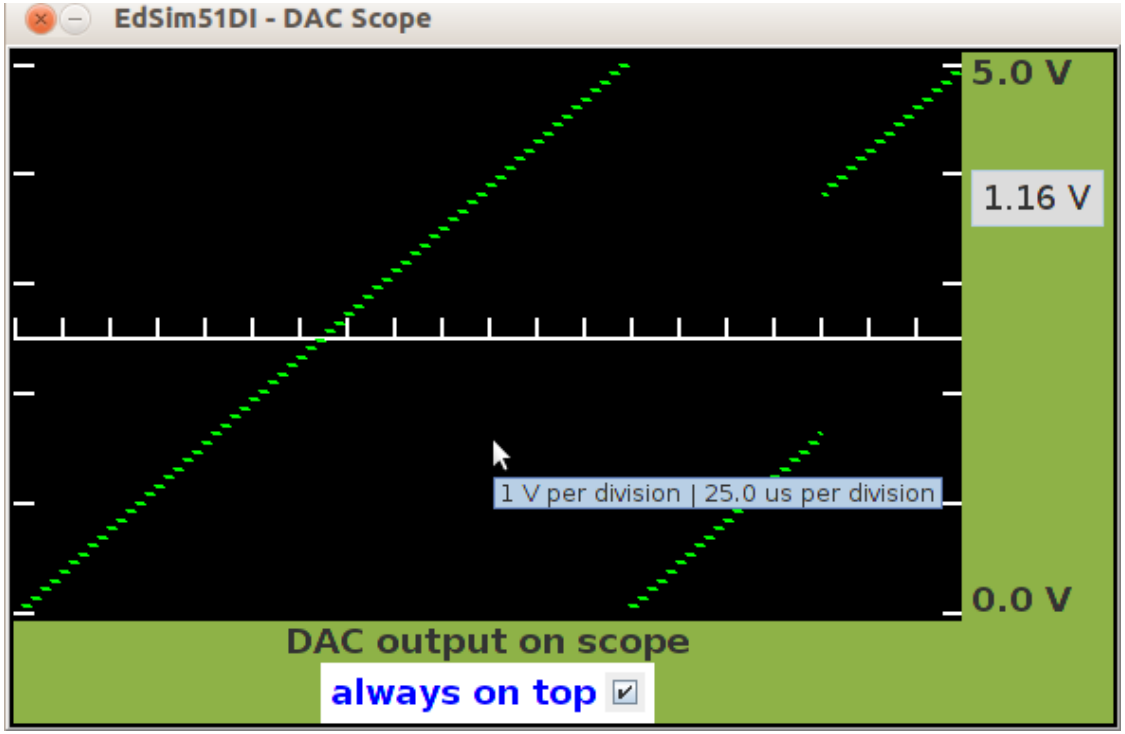
O 8051 não possui um DAC, mas no simulador EdSim51 tem um DAC externo.



DAC - Digital to Analog Converter – EdSim51

No EdSim51 o DAC está ligado no *scope*.

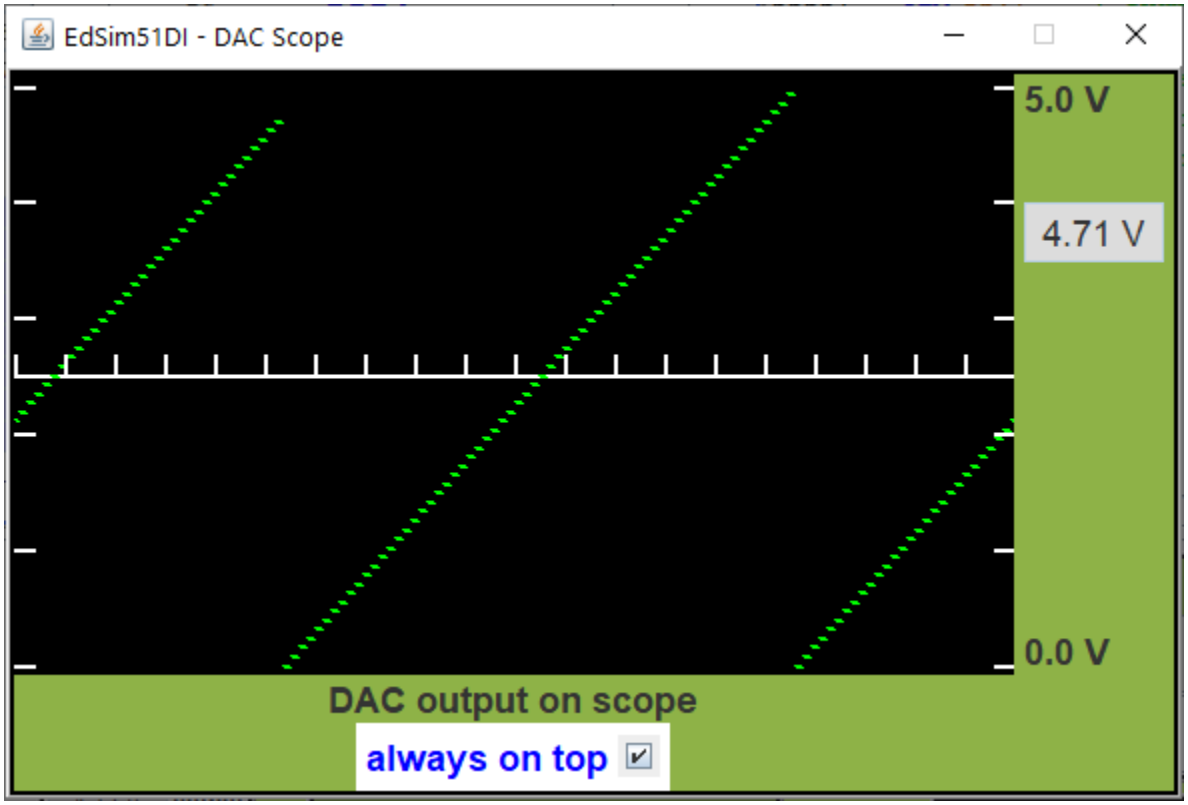




Programação

Exemplo de escrita no DAC

Este exemplo escreve no DAC, será escrito um valor que seja sempre incrementado.



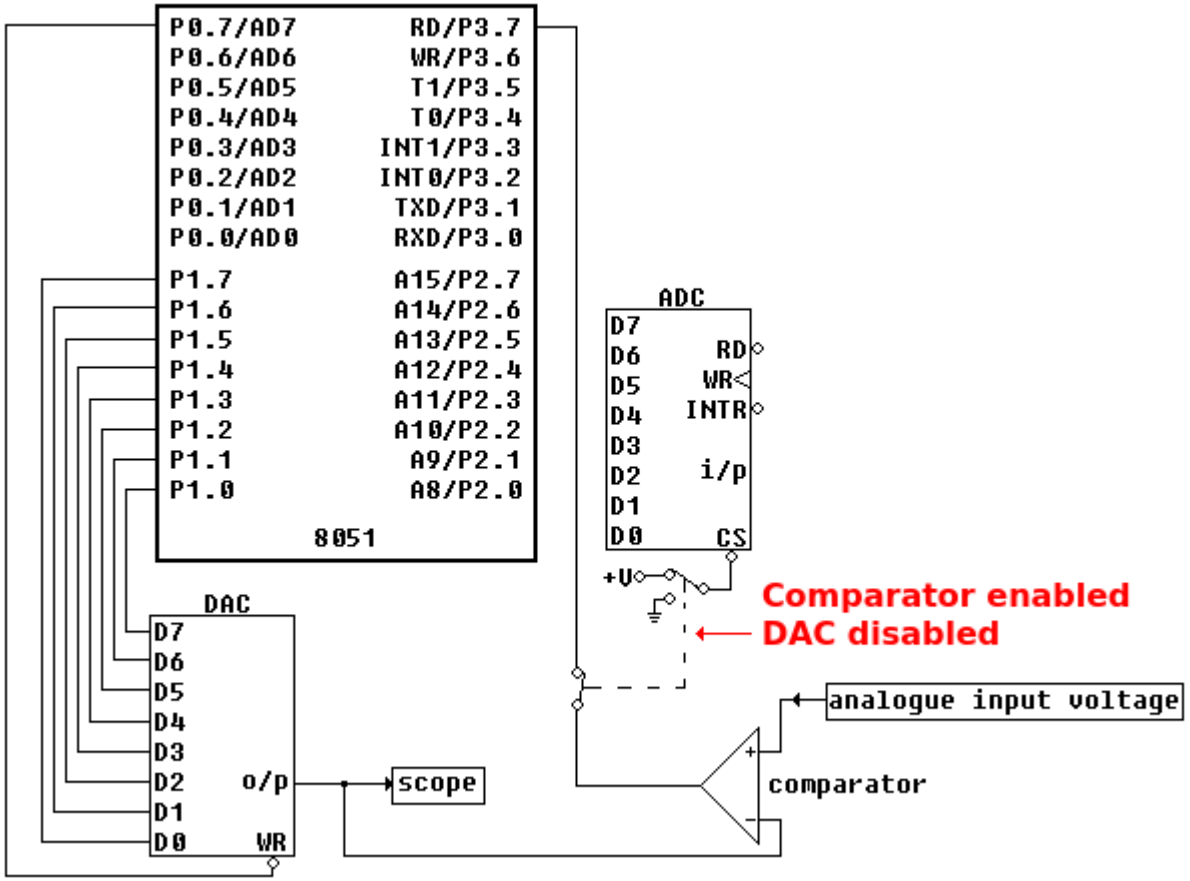
Exemplo de escrita no DAC

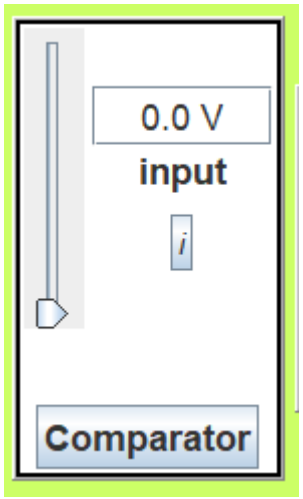
```
; This program generates a ramp on the DAC
; output.

; You can try adding values other than 8
; to the accumulator to see what this does
; to the ramp signal.

    CLR PØ.7     ; enable the DAC WR line
loop:
    MOV P1, A    ; move data in the accumulator to the ADC inputs (on P1)
    ADD A, #4    ; increase accumulator by 4
    JMP loop     ; jump back to loop
```

Exemplo 02

DAC - Digital to Analog Converter – EdSim51

Utilizando o DAC com o comparador.





Exemplo 02 de escrita no DAC e usando o comparador

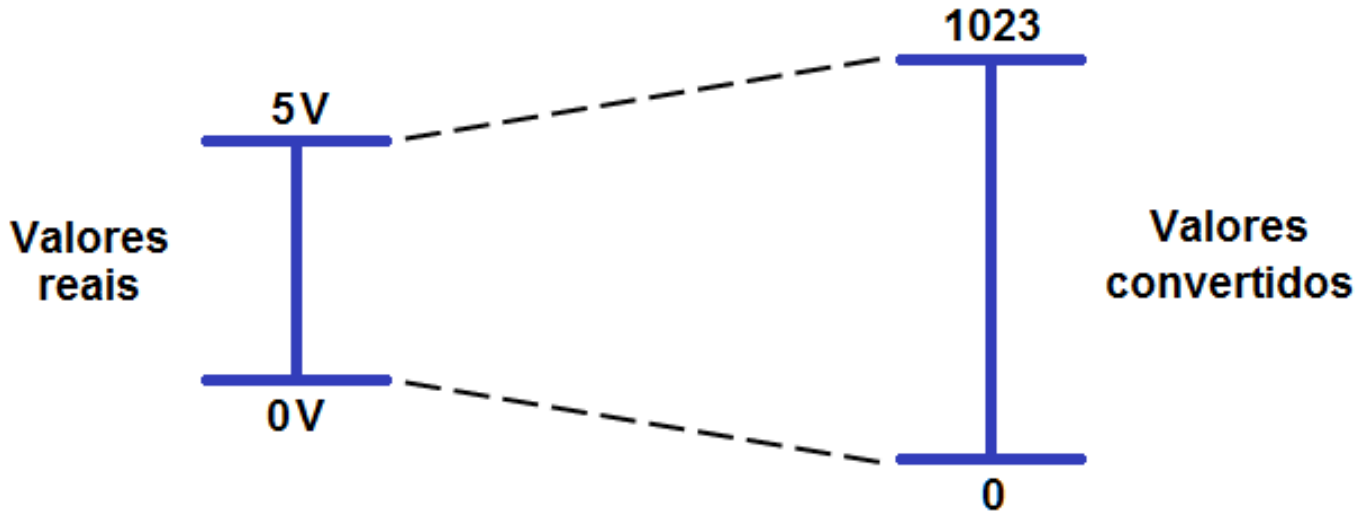
Execute o programa e observe o pino P3.7

```
; This program generates a ramp on the DAC
; output.

; You can try adding values other than 8
; to the accumulator to see what this does
; to the ramp signal.

    CLR PØ.7     ; enable the DAC WR line
loop:
    MOV P1, A    ; move data in the accumulator to the ADC inputs (on P1)
    ADD A, #4    ; increase accumulator by 4
    JMP loop     ; jump back to loop
```

ADC

ADC - Analog to Digital Converter

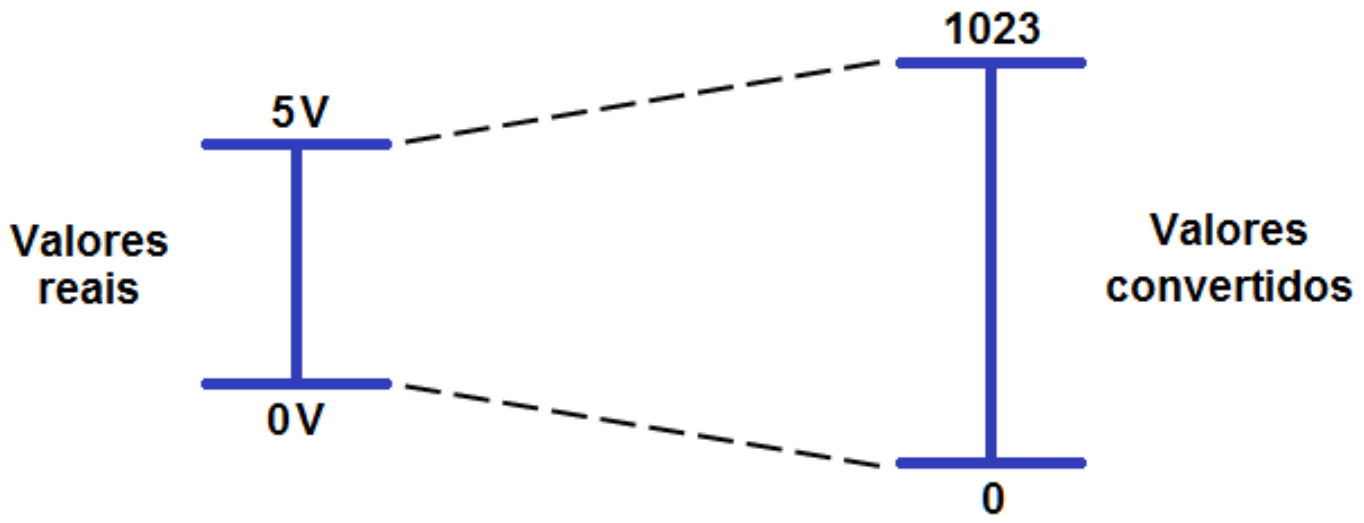
O conversor analógico-digital (ADC), é um circuito eletrônico que converte uma grandeza analógica (normalmente uma tensão ou uma corrente) em uma grandeza digital (código binário).



ADC - Analog to Digital Converter

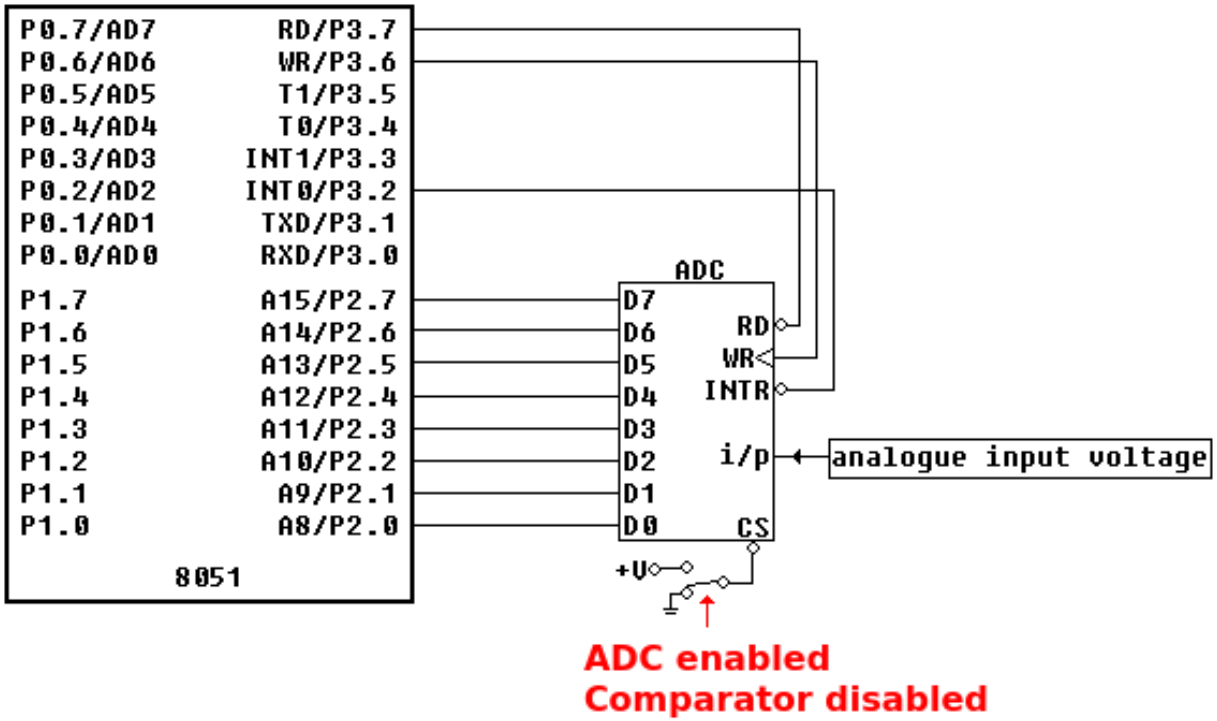
Exemplo:

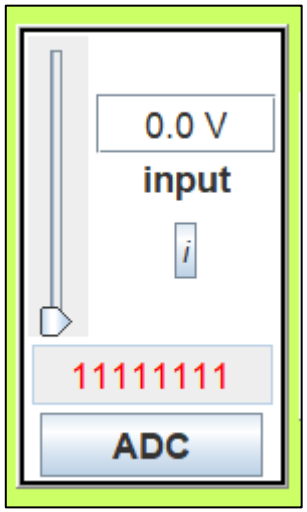
Podemos converter um sinal analógico de tensão 0 a 5V em números de 0 a 1023 em binário.



ADC - Analog to Digital Converter – EdSim51

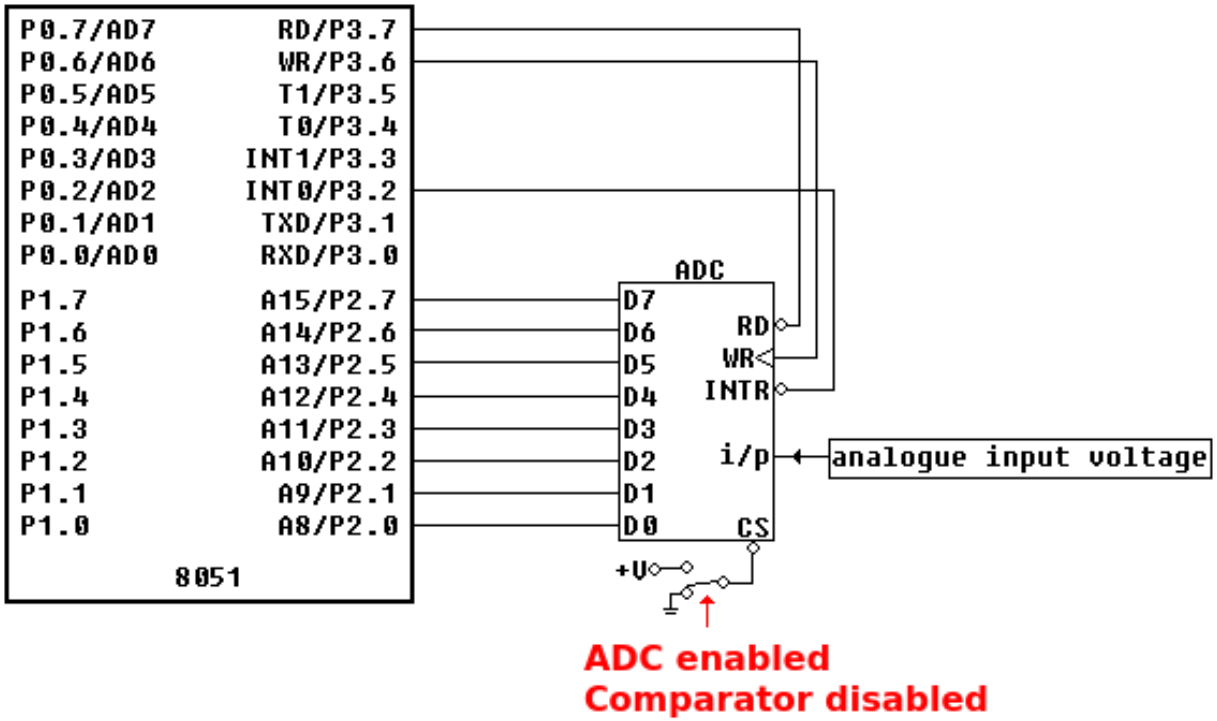
O 8051 não possui um ADC, mas no simulador EdSim51 tem um ADC externo.

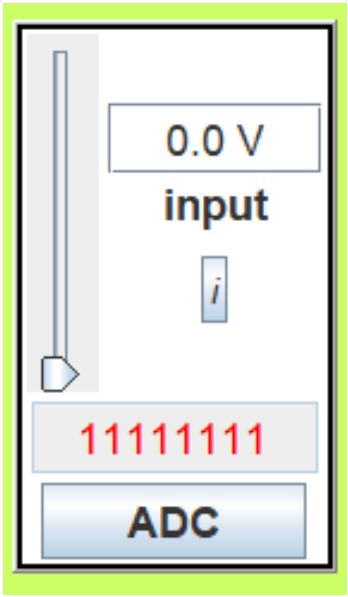




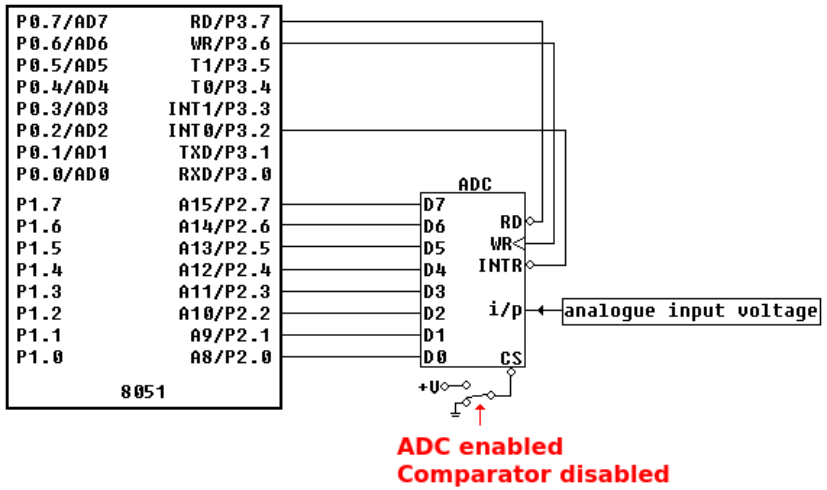
ADC - Analog to Digital Converter – EdSim51

Os valores do ADC devem ser modificados manualmente.





ADC - Analog to Digital Converter – EdSim51

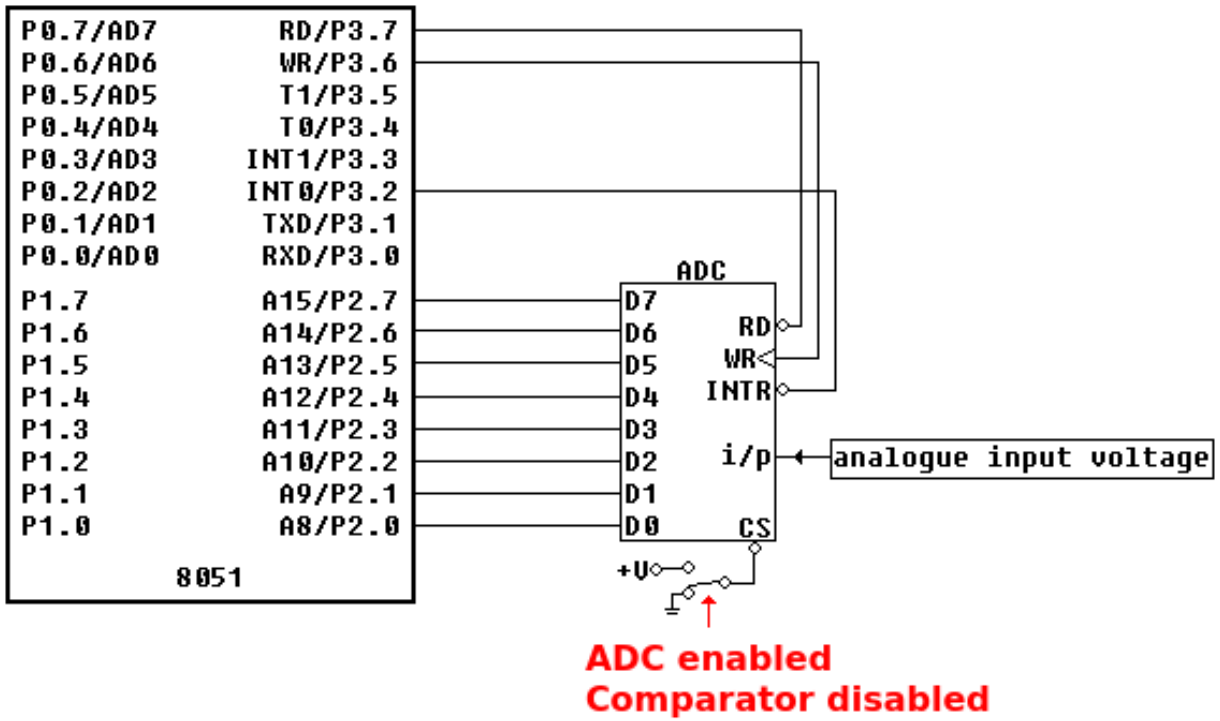


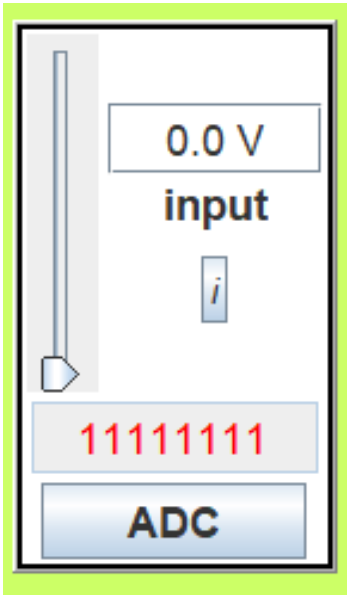
Pin	Function
RD	Enables the tri-state outputs, when logic 0.
WR	On a positive edge, initiates conversion.
INTR	Goes to logic 0 when conversion is complete and remains low until another conversion is initiated.
i/p	Analogue input signal applied here.
CS	Enables the device, when logic 0.
D0 - D7	Tri-state digital outputs.

Exemplo

Exemplo de leitura do ADC

Este exemplo escreve na porta P1 os valores que são lidos no ADC. O valor de tensão do ADC é modificado manualmente.





Exemplo de leitura do ADC

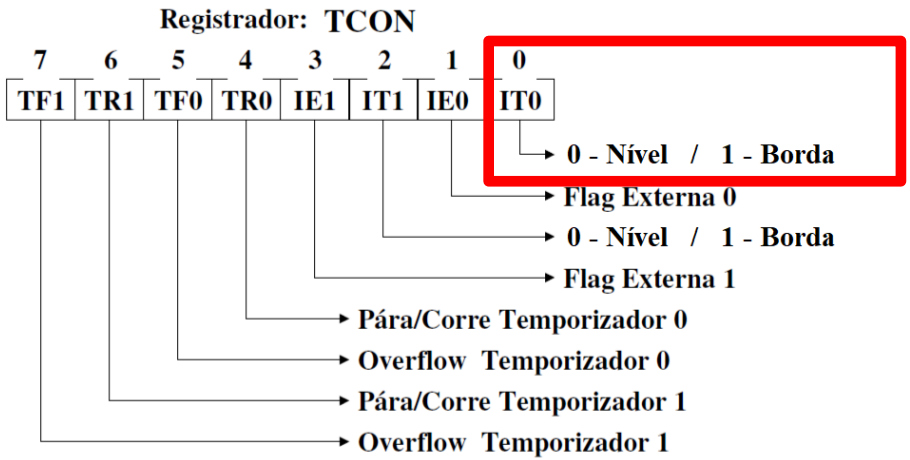
```
ORG Ø                ; reset vector
    JMP main         ; jump to the main program

ORG Ø3h              ; external Ø interrupt vector
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt

delay:
    MOV RØ, #4Ø
    DJNZ RØ, $
    RET
```

Exemplo de leitura do ADC

Pedido	Interrupção	Endereço
IE0	Externa 0	0003H
TF0	Temporizador 0	000BH
IE1	Externa 1	0013H
TF1	Temporizador 1	001BH
TI ou RI	Serial	0023H

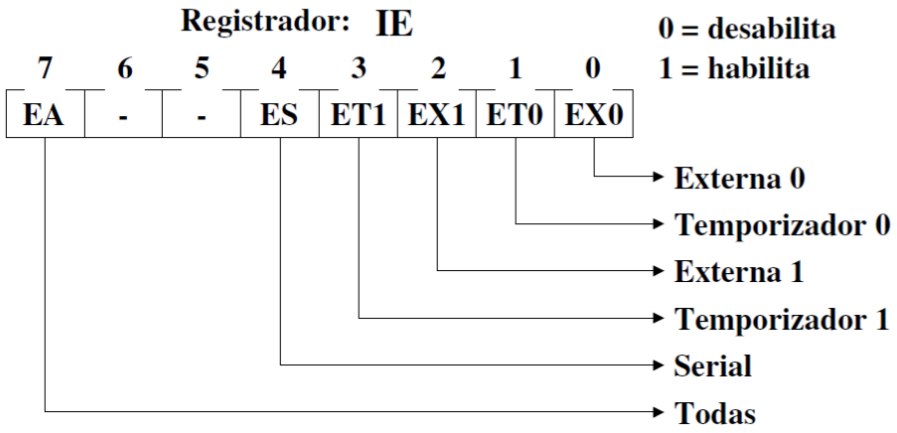
```
ORG Ø                ; reset vector
    JMP main         ; jump to the main program

ORG Ø3h              ; external Ø interrupt vector
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt
```

Exemplo de leitura do ADC




```
ORG Ø                ; reset vector
    JMP main         ; jump to the main program

ORG Ø3h              ; external Ø interrupt vector
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt
```

Exemplo de leitura do ADC




```
ORG Ø                ; reset vector
    JMP main         ; jump to the main program

ORG Ø3h              ; external Ø interrupt vector
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt
```

Exemplo de leitura do ADC

Pin	Function
RD	Enables the tri-state outputs, when logic 0.
WR	On a positive edge, initiates conversion.
INTR	Goes to logic 0 when conversion is complete and remains low until another conversion is initiated.
i/p	Analogue input signal applied here.
CS	Enables the device, when logic 0.
D0 - D7	Tri-state digital outputs.

```
ORG Ø                ; reset
    JMP main         ; jump t

ORG Ø3h              ; externa
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt
```

Exemplo de leitura do ADC

```
ORG Ø                ; reset vector
    JMP main         ; jump to the main program

ORG Ø3h              ; external Ø interrupt vector
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt
```

Exemplo de leitura do ADC

Pin	Function
RD	Enables the tri-state outputs, when logic 0.
WR	On a positive edge, initiates conversion.
INTR	Goes to logic 0 when conversion is complete and remains low until another conversion is initiated.
i/p	Analogue input signal applied here.
CS	Enables the device, when logic 0.
D0 - D7	Tri-state digital outputs.

```
ORG Ø                ; reset
    JMP main         ; jump t

ORG Ø3h              ; externa
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt
```

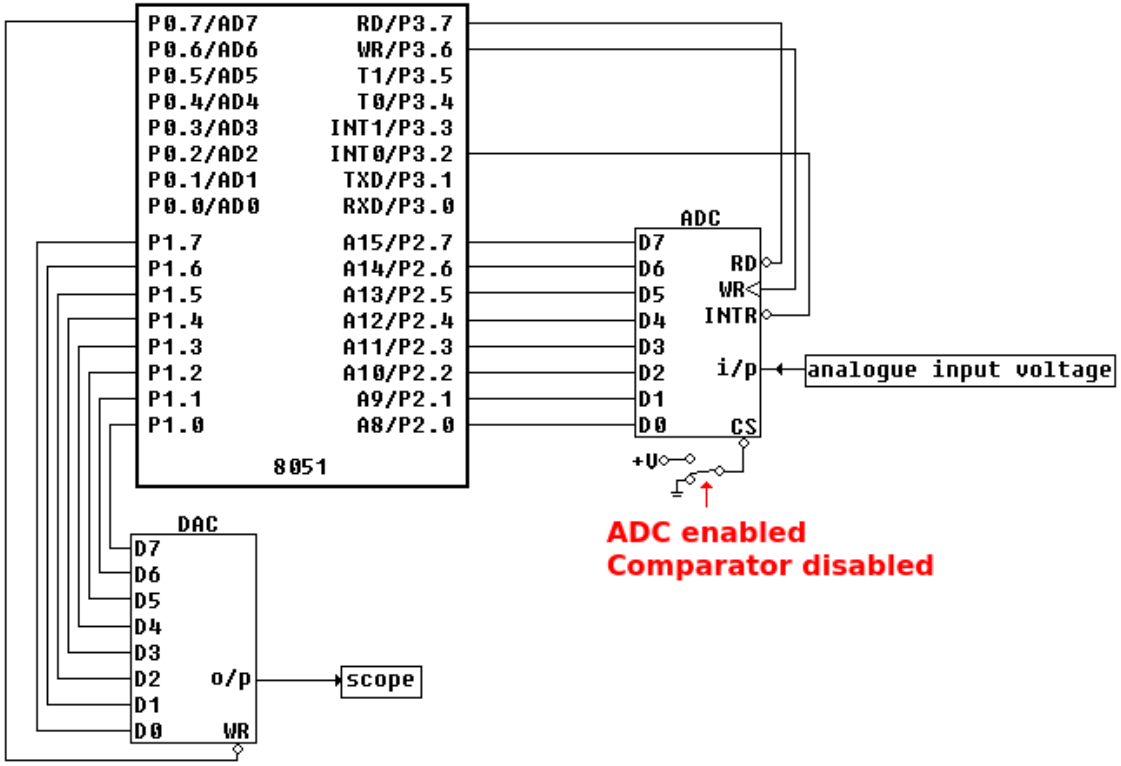
Exemplo de leitura do ADC

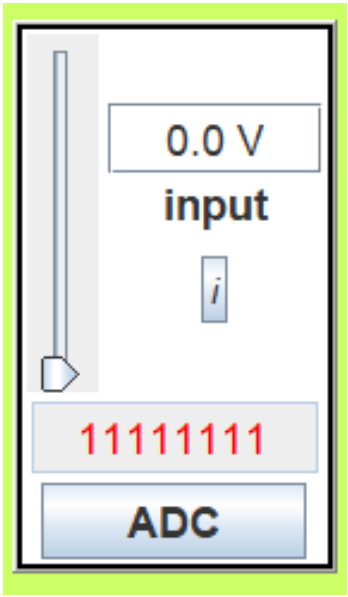
```
ORG Ø                ; reset vector
    JMP main         ; jump to the main program

ORG Ø3h              ; external Ø interrupt vector
    JMP extØISR      ; jump to the external Ø ISR

ORG 3ØH              ; main program starts here
main:
    SETB ITØ         ; set external Ø interrupt as edge-activated
    SETB EXØ         ; enable external Ø interrupt
    SETB EA          ; set the global interrupt enable bit
rot:
    CLR P3.6         ; clear ADC WR line
    SETB P3.6        ; then set it - this results in the required positive edge to start a conversion
    CALL delay
    JMP rot
; end of main program

; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7         ; clear the ADC RD line - this enables the data lines
    MOV P1, P2       ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7        ; disable the ADC data lines by setting RD
    RETI             ; return from interrupt

delay:
    MOV RØ, #4Ø
    DJNZ RØ, $
    RET
```

Exemplo 02

Exemplo de leitura do ADC

Este exemplo escreve no DAC os valores lidos no ADC. Utilizando o timer para calculo do tempo de leitura do ADC.





Exemplo 02 de leitura do ADC

```
ORG Ø               ; reset vector
    JMP main        ; jump to the main program

ORG 3               ; external Ø interrupt vector
    JMP extØISR     ; jump to the external Ø ISR

ORG ØBh             ; timer Ø interrupt vector
    JMP timerØISR   ; jump to timer Ø ISR

ORG 3ØH             ; main program starts here
main:
    SETB ITØ        ; set external Ø interrupt as edge-activated
    SETB EXØ        ; enable external Ø interrupt
    CLR PØ.7        ; enable DAC WR line
    MOV TMOD, #2        ; set timer Ø as 8-bit auto-reload interval timer

    MOV THØ, #-5Ø       ; | put -5Ø into timer Ø high-byte - this reload value,
                ; | with system clock of 12 MHz, will result in a timer Ø overflow every 5Ø us
    MOV TLØ, #-5Ø       ; | put the same value in the low byte to ensure the timer starts counting from
                ; | 236 (256 - 5Ø) rather than Ø

    SETB TRØ        ; start timer Ø
    SETB ETØ        ; enable timer Ø interrupt
    SETB EA         ; set the global interrupt enable bit
    JMP $           ; jump back to the same line (ie: do nothing)

; end of main program


; timer Ø ISR - simply starts an ADC conversion
timerØISR:
    CLR P3.6        ; clear ADC WR line
    SETB P3.6       ; then set it - this results in the required positive edge to start a conversion
    RETI            ; return from interrupt


; external Ø ISR - responds to the ADC conversion complete interrupt
extØISR:
    CLR P3.7        ; clear the ADC RD line - this enables the data lines
    MOV P1, P2      ; take the data from the ADC on P2 and send it to the DAC data lines on P1
    SETB P3.7       ; disable the ADC data lines by setting RD
    RETI            ; return from interrupt
```

Bibliografia

ZELENOVSKY, R.; MENDONÇA, A. Microcontroladores Programação e Projeto com a Família 8051. MZ Editora, RJ, 2005.

Gimenez, Salvador P. Microcontroladores 8051 - Teoria e Prática, Editora Érica, 2010.