Christian Ricardo Quelex

22695

Programacion de microcontroladores

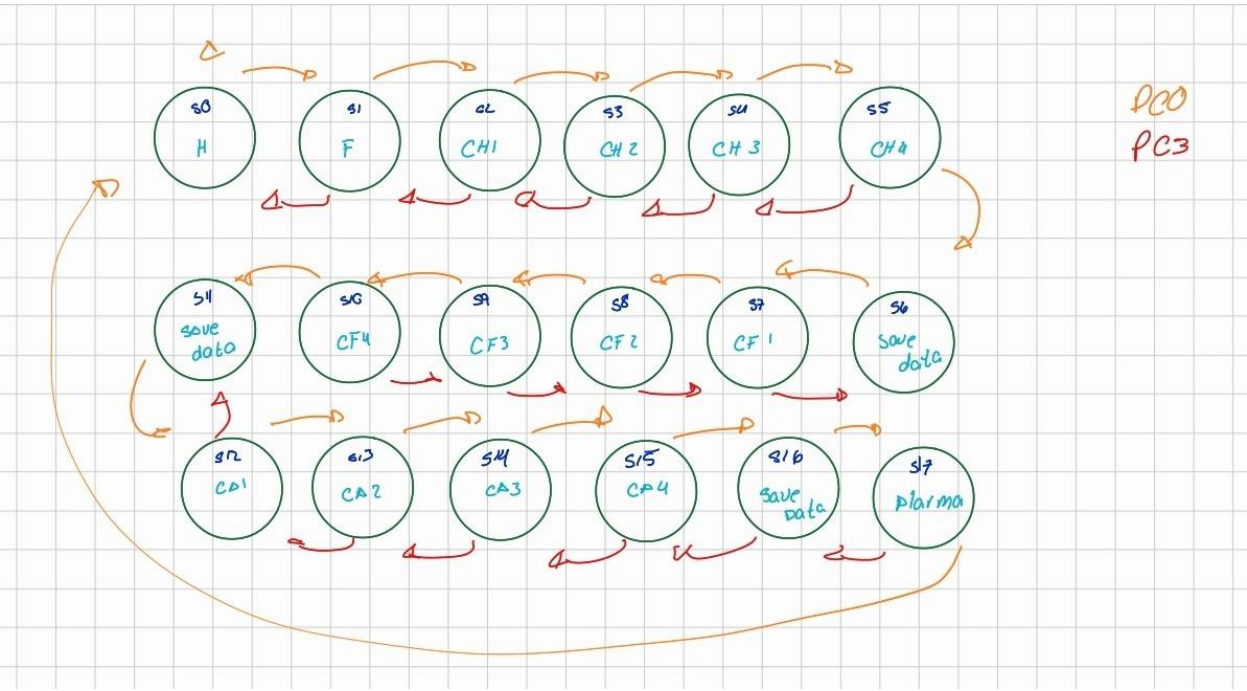
Proyecto 1

Reloj

Se realizo un reloj digital en la plataforma de Arduino, con el ATmega328p y en lenguaje ensamblador. Las funciones con las cuales cumple el reloj son las siguientes

1. Reloj en tiempo real
2. Despliega la fecha actual
3. Configuración de hora
4. Configuración de fecha
5. Configuración de alarma
6. Apagar alarma

FMS



Se utilizaron las interrupciones del timer0 y pin change de pcint1 (puerto C).

Calculos para timer0

n	8
Fclk	16Mhz
Prescaler	1024
Tmax	**16,4 mS**

valor desbordamiento	
n	8
Fclk	16Mhz
tdeseado	10 ms
Prescaler	1024
TCNT0	**99,75**

El valor de configuración para el overflow es de 99, lo que produce un tiempo muy cercano a 10 ms.

Explicación de Código

Debido a que el código es demasia largo únicamente colocare los fragmentos mas importantes. Algunos otros bloques son basicamente similares por lo que solo colocare uno y hare mención de cuantas veces se utilizo dentro del código y cual fue su propósito.

Configuración de Variables e Interrupciones

```
.DEF ESTADO = R18
.DEF COUNTER_T0 = R19
.DEF COUNTER = R20
.DEF MONTH = R24
.DEF DAY = R25
//.DEF INCMONTH = R26
//.DEF INCDAY = R27
.DEF FECHA = R21
//.DEF SUMA_DIAS = R30
.CSEG
.ORG 0X00
    JMP SETUP       ;RESET VECTOR
.ORG 0X0008         // Interrupciones en los puertos C
    JMP ISR_PCINT1
.ORG 0X0020
    JMP ISR_TIMER0  ;ISR: TIMER0 VECTOR
```

Configuración del Main

Se agrega configuración de registros indirectos y tablas de información

```
MAIN:

    LDI     R16, LOW(RAMEND)
    OUT     SPL, R16
    LDI     R17, HIGH(RAMEND)
    OUT     SPH, R17

    SEG: .DB 0x3F,0x06,0x5B,0x4F,0x66,0x6D,0x7D,0x07,0x7F,0x6F// ,//0x77,0x7C,0x39,0x5E,0x79,0x71

    DAYS: .DB 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31
```

Se habilitan los puertos de salida y entrada además de los puertos que van a ser utilizados para las interrupciones

```
SETUP:

    LDI     R16, 0x00       // deshabilitar puertos usart
    STS     UCSR0B, R16

    LDI     R16, 0xFF       // Salidas puertos D
    OUT     DDRD, R16
    LDI     R16, 0x00
    OUT     PORTD, R16

    LDI     R16, 0xFF
    OUT     DDRB, R16
    //SBI       DDRB, PB3
    //SBI       DDRB, PB4       // Salidas para multiplexiar
    //SBI       DDRB, PB5       // Luz a alarma - Buzzer

    LDI     R16, 0xF0       // Entradas Botones
    OUT     DDRC, R16
    LDI     R16, 0x0F
    OUT     PORTC, R16

    LDI     R21, 0              // configuracion de lista
    LDI     ZH, HIGH(SEG << 1)
    LDI     ZL, LOW(SEG << 1)
    ADD     ZL, R21
    LPM     R21, Z

    LDI     R16, (1 << PCINT8) | (1 << PCINT9) | (1 << PCINT10) | (1 << PCINT11)
    STS     PCMSK1, R16 // Puertos C como entras con interrupcion

    LDI     R16, (1 << PCIE1)
    STS     PCICR, R16  // interrupciones ping change en el puerto C

    CALL    INIT_T0
    SEI                 //Habilitar interrupciones globales

    CLR     ESTADO
    CLR     COUNTER_T0
    CLR     COUNTER
```

Se colocan valores iniciales para los registros de la SRAM y algunos otros registro de uso común. Son muchos mas pero es la misma idea para todos

```
LDI     R16, 0
STS     0x0121, R16         //Display1 Segundos
LDI     R16, 5
STS     0x0122, R16         //Display2 Decenas S
LDI     R16, 9
STS     0x0123, R16         //Display3 Min
LDI     R16, 5
STS     0x0124, R16         //Display4 Decena M
LDI     R16, 3 // 3
STS     0x0125, R16         //Display5 Horas
LDI     R16, 2
STS     0x0126, R16         //Display6 Decenas H
CLR     R16
STS     0x0127, R16         //Estado de display


LDI     R16, 2
STS     0x0151, R16         //Display1 DIA
LDI     R16, 1
STS     0x0152, R16         //Display2 DECENA DE DIA
LDI     R16, 1
STS     0x0153, R16         //Display3 MES
LDI     R16, 3
STS     0x0154, R16         //Display4 DECENA DE MES
//LDI       R16, 3 // 3
//STS       0x0155, R16         //Display5 Horas
//LDI       R16, 2
//STS       0x0156, R16         //Display6 Decenas H
//CLR       R16
//STS       0x0157, R16         //Estado de display

// REGISTROS PARA COMPARACION DE FECHA
CLR R2
CLR R3
CLR R4
CLR R6
CLR R7      // USO PARA MESES
```

El loop principal contiene todos los estados y sirve únicamente para comprar en que estado nos encontramos

```
LOOP:       //Listado de Estados

    //LDS       R18, STATE_ADDR
    CPI     ESTADO, 0
    BREQ    ESTADO0PASO
    CPI     ESTADO, 1
    BREQ    ESTADO1PASO
    CPI     ESTADO, 2
    BREQ    ESTADO2PASO
    CPI     ESTADO, 3
    BREQ    ESTADO3PASO
    CPI     ESTADO, 4
    BREQ    ESTADO4PASO
    CPI     ESTADO, 5
    BREQ    ESTADO5PASO

    CPI     ESTADO, 6
    BREQ    ESTADO6PASO
    CPI     ESTADO, 7S
    BREQ    ESTADO7PASO
    CPI     ESTADO, 8
    BREQ    ESTADO8PASO
    CPI     ESTADO, 9
    BREQ    ESTADO9PASO
    CPI     ESTADO, 10
    BREQ    ESTADO10PASO
    CPI     ESTADO, 11
    BREQ    ESTADO11PASO

    CPI     ESTADO, 12
    BREQ    ESTADO12PASO
    CPI     ESTADO, 13
    BREQ    ESTADO13PASO
    CPI     ESTADO, 14
    BREQ    ESTADO14PASO
    CPI     ESTADO, 15
    BREQ    ESTADO15PASO
    CPI     ESTADO, 16
    BREQ    ESTADO16PASO
    CPI     ESTADO, 17
    BREQ    ESTADO17PASO
```

Existen algunos estados de paso que sirven para realizar saltos grandes dado que los BREQ no son capaces de avanzar grandes distancias entre instrucciones

```
ESTADO0PASO:        // MOSTRAR HORA
    JMP     ESTADO0

ESTADO1PASO:        // MOSTRAR FECHA
    JMP     ESTADO1
ESTADO2PASO:
    CLR     R16
    STS     0x0132, R16     //ESTADO DISPLAY2
    STS     0x0133, R16     //ESTADO DISPLAY3
    STS     0x0134, R16     //ESTADO DISPLAY4
    JMP     ESTADO2
ESTADO3PASO:
    CLR     R16
    STS     0x0133, R16     //ESTADO DISPLAY3
    STS     0x0134, R16     //ESTADO DISPLAY4
    JMP     ESTADO3
ESTADO4PASO:
    CLR     R16
    STS     0x0134, R16     //ESTADO DISPLAY4
    JMP ESTADO4

ESTADO5PASO:
    JMP     ESTADO5
ESTADO6PASO:
    JMP     ESTADO6
ESTADO7PASO:
    CLR     R16
    STS     0x0162, R16
    STS     0x0163, R16
    STS     0x0164, R16
    JMP     ESTADO7
ESTADO8PASO:
    CLR     R16
    STS     0x0164, R16
    JMP     ESTADO8
ESTADO9PASO:
```

Dentro del estado contamos con presentaciones de información para los displays

```
ESTADO0:        // Estado Base - Muestra la Hora

    CALL ACARREO_DIAS
    CALL SECONDS_DISPLAYS
    // PRIMER DISPLAY1
    CALL CLEAN
    LDS R22, 0x0123
    CALL DISPLAY
    LDI R16, 0x02
    OUT PORTB, R16      //ENCENCER CIERTO TRANSISTOR
    OUT PORTD, R22      //CARGA VALORES DE DISPLAY

    //SEGUNDO DISPLAY2
    CALL CLEAN
    LDS R22, 0x0124
    CALL DISPLAY
    LDI R16, 0x01
    OUT PORTB, R16      //ENCENDER CIERTO TRANSISTOR
    OUT PORTD, R22      //CARGAR VALORES DE DISPLAY


    // TERCER DISPLAY HORAS1
    CALL CLEAN
    LDS R22, 0x0125
    CALL DISPLAY
    LDI R16, 0x20
    OUT PORTB, R16
    OUT PORTD, R22

    // CUARTO DISPLAY HORAS2
    CALL CLEAN
    LDS R22, 0x0126
    CALL DISPLAY
    CBI PORTC, PC4
    LDI R16, 0x10
    OUT PORTB, R16
    OUT PORTD, R22


    CALL CLEAN
    LDI R16, 0x00
    SBI PORTC, PC4
    OUT PORTB, R16
    LDI R22, 0b01110110
```

Existen algunas cadenas lógicas que lo que hacen es limitar los contadores para el correcto funcionamiento del overflow tanto de las horas como de la fecha.

```
ZEROS:                          // Conador de decenas
    LDI     R22, 0
    STS 0x0121, R22

    LDS R22, 0x0122
    INC R22
    STS 0x0122, R22
    CPI R22, 6
    BREQ SIXS
    RJMP LOOP

SIXS:
    LDI R22, 0
    STS 0x0122, R22

    LDS R22, 0x0123
    INC R22
    STS 0x0123, R22
    CPI R22, 10
    BREQ ZEROM
    RJMP LOOP


ZEROM:                          // Conador de decenas
    LDI     R22, 0
    STS 0x0123, R22

    LDS R22, 0x0124
    INC R22
    STS 0x0124, R22
    CPI R22, 6
    BREQ SIXM
    RJMP LOOP

SIXM:
    LDI R22, 0
    STS 0x0124, R22

    LDS R22, 0x0125
    INC R22
    STS 0x0125, R22
    CPI R22, 10
```

Existen algunas otras sub rutinas que únicamente se encargan de presentar el estado actual de los display para la visualización y la ayuda de modificación de los dígitos en tiempo real.

```
MUX_DISPLAYS:
    CALL SECONDS_DISPLAYS

    CALL CLEAN

    LDS R22, 0x0131
    CALL DISPLAY
    LDI R16, 0x02
    OUT PORTB, R16
    OUT PORTD, R22

    CALL CLEAN

    LDS R22, 0x0132
    CALL DISPLAY
    LDI R16, 0x01
    OUT PORTB, R16
    OUT PORTD, R22

    CALL CLEAN

    LDS R22, 0x0133
    CALL DISPLAY
    LDI R16, 0x20
    OUT PORTB, R16
    OUT PORTD, R22

    CALL CLEAN

    LDS R22, 0x0134
    CALL DISPLAY
    LDI R16, 0x10
    OUT PORTB, R16
    OUT PORTD, R22

    CALL SHOW_STATE_SC

    SBI PIND, PD7
    RET
```

Instrucciones de limpieza y configuración de presentaciones de valores de display

```
CLEAN:

    CLR R16
    OUT PORTB, R16
    OUT PORTD, R16
    CBI PORTC, PC4
    RET

DISPLAY:

    LDI ZH, HIGH(SEG << 1)
    LDI ZL, LOW(SEG << 1)
    ADD ZL, R22
    LPM R22, Z
    RET
```

Esta es una subrutina encarga de los limites y overflow de los meses

Las interrupciones también tienen estados de paso para la modificación dependiendo del estado en el cual nos encontremos

```
ISR_PCINT1:
    PUSH R16
    IN R16, SREG
    PUSH R16


    CPI ESTADO, 0
    BREQ ESTADO0_ISR_P
    CPI ESTADO, 1
    BREQ ESTADO1_ISR_P
    CPI ESTADO, 2
    BREQ ESTADO2_ISR_P
    CPI ESTADO, 3
    BREQ ESTADO3_ISR_P
    CPI ESTADO, 4
    BREQ ESTADO4_ISR_P
    CPI ESTADO, 5
    BREQ ESTADO5_ISR_P


    //CPI ESTADO, 6             // AHORA AQUI ESTA ES
    //BREQ ESTADO6_ISR_P
    CPI ESTADO, 7
    BREQ ESTADO7_ISR_P
    CPI ESTADO, 8
    BREQ ESTADO8_ISR_P
    CPI ESTADO, 9
    BREQ ESTADO9_ISR_P
    CPI ESTADO, 10
    BREQ ESTADO10_ISR_P
    //  CPI ESTADO, 11
//  BREQ ESTADO11_ISR_P

    CPI ESTADO, 12
    BREQ ESTADO12_ISR_P
    CPI ESTADO, 13
```

Dentro de los estados de interrupción se leen los pines destinados a la interrupción y dependiendo de cual sea activado realiza distintos cambios. Por defecto el PC0 incrementa el estado, PC1 y PC2 incrementan o decrementan el valor de los registros utilizados para guardar el valor de los displays, y el PC3 sirve para decrementar estados.

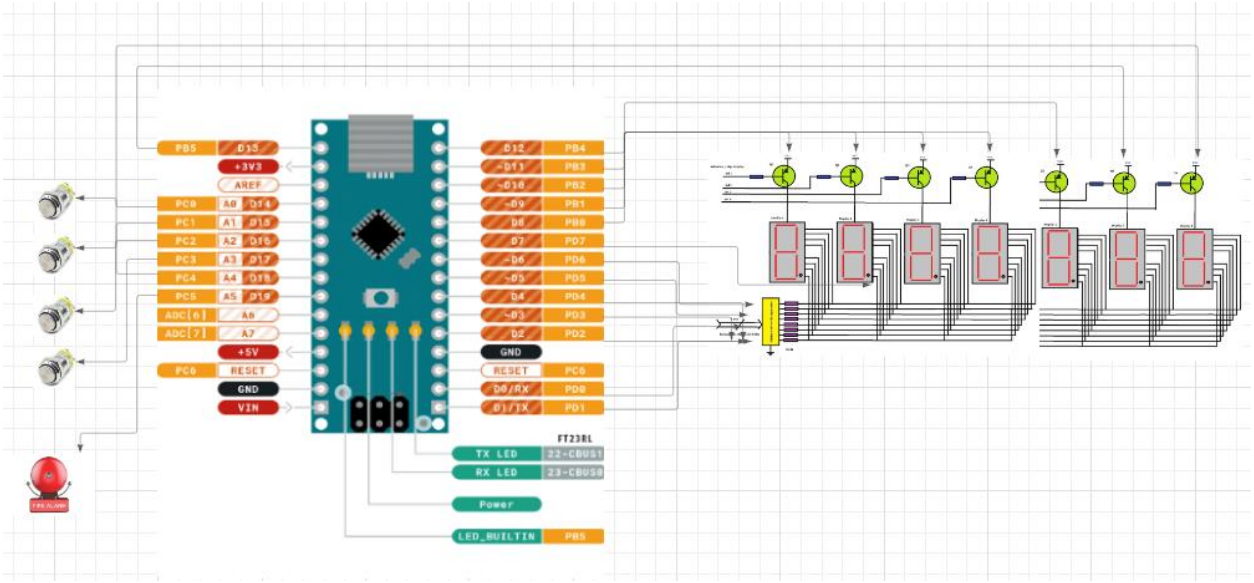
```
ESTADO0_ISR:

    IN R16, PINC
    SBRS R16, 1
    NOP
    SBRS R16, 2
    NOP
    SBRS R16, 0
    CLR COUNTER
    SBRS R16, 0
    LDI ESTADO, 1
    SBRS R16, 3
    NOP

    RJMP ISR_POP


ESTADO1_ISR:

    IN R16, PINC
    SBRS R16, 1
    NOP
    SBRS R16, 2
    NOP
    SBRS R16, 0
    CLR COUNTER
    SBRS R16, 0
    LDI ESTADO, 2
    SBRS R16, 3
    LDI ESTADO, 0

    RJMP ISR_POP
```

Contamos con las subrutinas encargadas de guardar y regresar los valores globales luego de las interrupciones. Configuración de prescaler para la configuración interrupciones y el valor inicial para el desbordamiento e interrupción de desbordamiento del timer0.

```
ISR_POP:
    SBI PCIFR, PCIF1
    POP R16
    OUT SREG, R16
    POP R16
    RETI


    // prescaler para timer0
INIT_T0:
    LDI     R16, (1 << CS02) | (1 << CS00)
    OUT     TCCR0B, R16     // prescaler de 1024

    LDI     R16, 99         // valor de overflow
    OUT     TCNT0, R16      // cargar el valor de overflow

    LDI     R16, (1 << TOIE0)
    STS     TIMSK0, R16         // habilitar interrupcion
    RET
    // Carga de Valores
ISR_TIMER0:
    LDI     R16, 99         // valor de timer
    OUT     TCNT0, R16      // cargar valor de overflow
    SBI     TIFR0, TOV0     // Apagar bandera
    INC     COUNTER_T0              // Incrementar contador cada 10ms
    RETI
```

Puertos utilizados

Función					Función
MUX 6	PB5	D13	D12	PB4	MUX 5
	3V3		D11	PB3	MUX 4
	REF		D10	PB2	MUX 3
INC E	PC0	A0	D9	PB1	MUX 2
INC C	PC1	A1	D8	PB0	MUX 1
DEC C	PC2	A2	D7	PD7	LED S
DEC E	PC3	A3	D6	PD6	G
MUX 7	PC4	A4	D5	PD5	F
ALARMA	PC5	A5	D4	PD4	E
	ADC 6	A6	D3	PD3	D
	ADC 7	A7	D2	PD2	C
	5V		GND		
	PD6	RST	RST	PC6	
	GND		RX	PD0	A
	VIN		TX	PD1	B



Link Progra

https://youtu.be/7_YJUa0FaDE

Link Funcionamiento

https://youtu.be/iy5InU7u5Io

Link GitHub

https://github.com/ChristianQuelex/proyectos/tree/main/Proyecto%201%20Micros%2022695