

TECNOLÓGICO NACIONAL DE MÉXICO.

INSTITUTO TECNOLÓGICO SUPERIOR DE TEPOSCOLULA.

INGENIERÍA EN SISTEMAS COMPUTACIONALES.

SISTEMAS PROGRAMABLES.

REPORTE DE PROYECTO SEMESTRAL:

ELABORACIÓN DE UN RELOJ DESPERTADOR.

**ELABORADO POR:**

**-BENÍTEZ NAVA FRANCISCO EDUARDO.**

**-HERNÁNDEZ MARTÍNEZ OSCAR.**

**-JUÁREZ PAZ ELBA.**

**-MORALES CARRASCO ALAN DE JESÚS.**

**-REGINO ÁLVAREZ MAURICIO.**

SEMESTRE: 7° GRUPO: “A”

DOCENTE:

ING. MIGUEL SIBAJA CRUZ

PERIODO:

AGOSTO – DICIEMBRE 2018

SAN PEDRO Y SAN PABLO TEPOSCOLULA, OAXACA.









INTRODUCCIÓN.

# INTRODUCCIÓN

La tecnología va en un gran aumento, y evidentemente no planea detenerse, vivimos en un mundo donde cualquier cosa ya utiliza las últimas innovaciones para que sea más efectivo al realizar las operaciones pertinentes, sistemas en los cuales todo puedo controlarse desde una distancia mucho mayor con la ayuda del internet, y al mismo tiempo, con la ayuda de un solo botón que esté perfectamente programado se hacen múltiples tareas desde cualquier lugar del mundo.

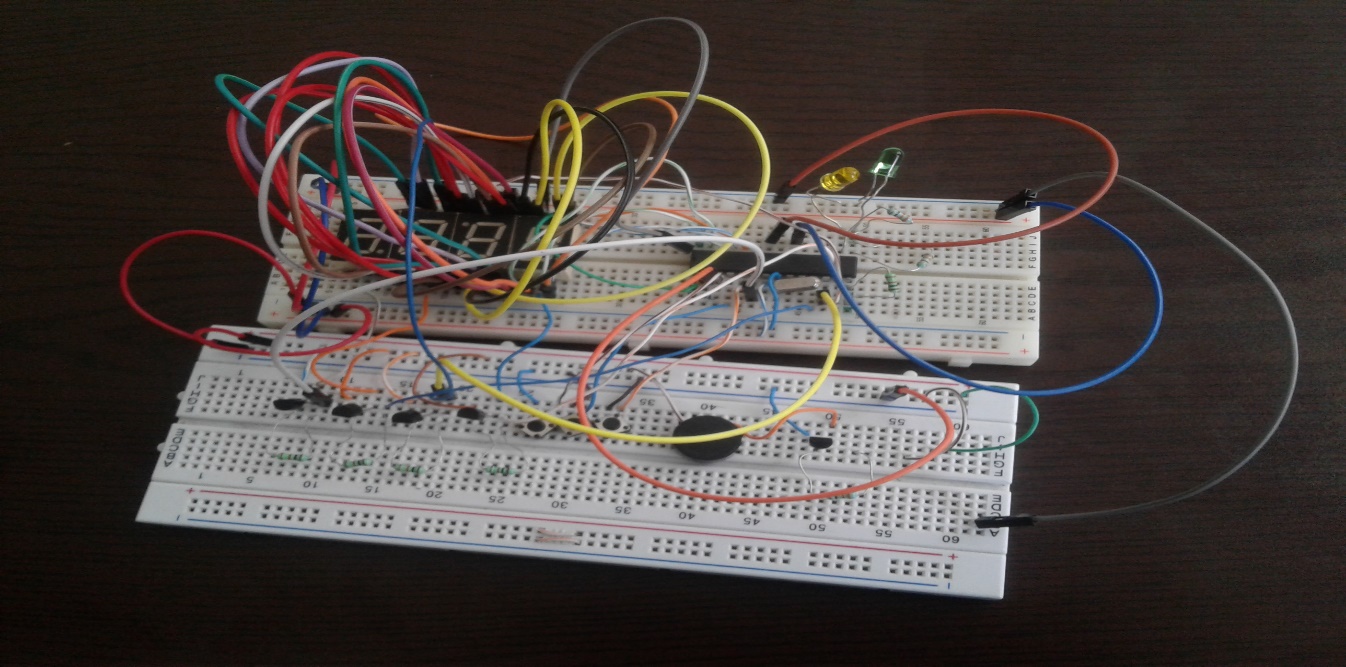
Siguiendo hablando de los grandes avances, todo sistema aún más sofisticado surgió cuando lograron implementar lenguajes de programación en cosas de electrónica y circuitos, que, combinados, hacen grandes maravillas, se hacen todo de creaciones y mejoras que cuando se usan parecen muy sencillas, pero detrás de todo eso, se esconde cientos y cientos de líneas de código, pero junto a eso, en la penumbra y escondida de la vista de todo usuario, un laberinto de cables, resistencias, leds, entre otros componentes, que juntos, harán funcionar algo tan básico como lo es prender, apagar y configurar un despertador común. Y es precisamente eso, lo que nos trae aquí hoy, pues durante las páginas siguientes, se hablará de cómo fue construyéndose este proyecto y todo lo que implicó para su resultado final.

# DESARROLLO DEL PROYECTO.

## Diseño

El diseño del despertador se ha tomado del libro “Los micro controladores avr de atmel”



En la realización de este proyecto, se construyó un prototipo de reloj despertador,

Para lograr esto, se necesitó de ciertos materiales que fueron de gran utilidad para llegar a su construcción, los cuales fueron:

-2 protoboards.

-4 resistencias.

-3 Botones

-1 zumbador.

-2 leds.

-4 displays de ánodo común.

-1 Atmega8.

-Resistencias de distintos voltajes.

-1 Cristal de cuarzo.

-1 interruptor.

-Jumpers macho – macho.

-Cable UTP.

Cada uno de estos materiales juega un rol muy específico en el funcionamiento de todo el circuito, pues el propósito principal es que al marcar una hora en específico pueda alertarnos de que ya es la hora marcada, tal y como lo haría un despertador convencional.

Cabe mencionar también que, para su funcionamiento debido y adecuado, se tuvo que programar un código en específico en el lenguaje de programación C para que realizara sus funciones de una manera adecuada.

|  |
| --- |
| /\*  \* GccApplication1.cpp  \*  \* Created: 04/12/2018 06:31:58 p. m.  \* Author : Z705-mm\_13  \*/  #include <avr/io.h>  #include <avr/pgmspace.h>  #include <avr/iom8.h>  #include <avr/interrupt.h>  #include <avr/delay.h>  /\*\*  \* Se omiten las declaraciones de variables globales  \* y de bibliotecas de funciones.  \*/  *int* h\_act, m\_act, s\_act;  *int* h\_alrm, m\_alrm;  *int* modo;  *int* AM\_F, AM\_F\_A;  *int* ALARM;  *int* CAMBIO;  *int* Disp[4];  *int* Sg7[10];  *int* habs[4];  *void* ubica() { // Ubica los datos a mostrar en un arreglo global     switch(modo) { // Segon sea el modo        case 1: Disp[3] = pgm\_read\_byte(&Sg7[h\_act/10]);// Hora Actual        Disp[2] = pgm\_read\_byte(&Sg7[h\_act%10]);        Disp[1] = pgm\_read\_byte(&Sg7[m\_act/10]);        Disp[0] = pgm\_read\_byte(&Sg7[m\_act%10]);        if( CAMBIO ) // Dos puntos parpadeando         PORTB = PORTB | 0x01;        else         PORTB = PORTB & 0xFE;        break;        case 2: Disp[3] = 0xFF; // Segundos        Disp[2] = 0xFF;        Disp[1] = pgm\_read\_byte(&Sg7[s\_act/10]);        Disp[0] = pgm\_read\_byte(&Sg7[s\_act%10]);        break;        case 3: if( CAMBIO ) { // Hora actual           Disp[3] = pgm\_read\_byte(&Sg7[h\_act/10]);// parpadeando           Disp[2] = pgm\_read\_byte(&Sg7[h\_act%10]);           } else {           Disp[3] = 0xFF;           Disp[2] = 0xFF;        }        Disp[1] = pgm\_read\_byte(&Sg7[m\_act/10]);// Minutos actuales        Disp[0] = pgm\_read\_byte(&Sg7[m\_act%10]); // normal        break;        case 4: Disp[3] = pgm\_read\_byte(&Sg7[h\_act/10]); //Hora actual normal        Disp[2] = pgm\_read\_byte(&Sg7[h\_act%10]);        if( CAMBIO ) { // Minutos actuales           Disp[1] = pgm\_read\_byte(&Sg7[m\_act/10]);// parpadeando           Disp[0] = pgm\_read\_byte(&Sg7[m\_act%10]);           } else {           Disp[1] = 0xFF;           Disp[0] = 0xFF;        }        break;        case 5: if( CAMBIO ) { // Hora de la alarma           Disp[3] = pgm\_read\_byte(&Sg7[h\_alrm/10]);//parpadeando           Disp[2] = pgm\_read\_byte(&Sg7[h\_alrm%10]);        }        else {           Disp[3] = 0xFF;           Disp[2] = 0xFF;        }        Disp[1] = pgm\_read\_byte(&Sg7[m\_alrm/10]);// Minutos de la        Disp[0] = pgm\_read\_byte(&Sg7[m\_alrm%10]);// alarma normal        break;        case 6: Disp[3] = pgm\_read\_byte(&Sg7[h\_alrm/10]);//Hora de la alarma        Disp[2] = pgm\_read\_byte(&Sg7[h\_alrm%10]); // normal        if( CAMBIO ) {           Disp[1] = pgm\_read\_byte(&Sg7[m\_alrm/10]);//Minutos de la           Disp[0] = pgm\_read\_byte(&Sg7[m\_alrm%10]);//alarma parpadeando           } else {           Disp[1] = 0xFF;           Disp[0] = 0xFF;        }        break;     }  }  *void* mostrar() { // Funcion para el despliegue de datos  *unsigned* *char* i;  ubica(); // Coloca la informacion a mostrar  for( i = 0; i < 4; i++) {  PORTC = Disp[i]; // Envoa el dato al puerto  PORTD = pgm\_read\_byte(&habs[i]); // Habilita para su despliegue  \_delay\_ms(5); // Espera se vea adecuadamente  PORTD = 0xFF; // Inhabilita para no introducir ruido  } // en otro display  }  ISR(TIMER2\_OVF\_vect) { // Se ejecuta cada medio segundo  if( CAMBIO ) {  CAMBIO = 0;  s\_act++; // Incrementa los segundos  if( s\_act == 60 ) { // Si son 60 incrementa los minutos  s\_act = 0;  m\_act++;  if( ALARM ) {  ALARM = 0; // Si la alarma esto activa, la apaga  PORTB = PORTB & 0b11110111; // por ser otro minuto  }  if(m\_act == 60) { // Si son 60 minutos incrementa las horas  m\_act = 0;  h\_act++;  if( h\_act == 13) // Con la hora 13 se regresa a 1  h\_act = 1;  if( h\_act == 12 ) { // Con la hora 12 se ajusta la andera  if( AM\_F == 1 ) { // AM\_F y las salidas AM y FM  AM\_F = 0;  PORTB = PORTB & 0b11111101;  PORTB = PORTB | 0b00000100;  } else {  AM\_F = 1;  PORTB = PORTB & 0b11111011;  PORTB = PORTB | 0b00000010;  }  }  }  // En cada nuevo minuto tambion revisa si se debe activar la alarma  if( !(PIND&0x01) && AM\_F==AM\_F\_A && h\_act==h\_alrm && m\_act==m\_alrm ) {  ALARM = 1;  PORTB = PORTB | 0b00001000;  }  }  } else // Modifica la bandera si no es el  // segundo  CAMBIO = 1; // completo  }  ISR(INT0\_vect) { // Atiende al boton UP  switch( modo ) {  case 3: h\_act++; // Incrementa hora actual  if( h\_act == 13 ) // De 13 pasa a 1  h\_act = 1;  if( h\_act == 12 ) { // En 12 ajusta bandera AM\_F  if( AM\_F ) { // y salidas AM y PM  AM\_F = 0;  PORTB = PORTB & 0b11111101;  PORTB = PORTB | 0b00000100;  } else {  AM\_F = 1;  PORTB = PORTB & 0b11111011;  PORTB = PORTB | 0b00000010;  }  }  break;  case 4: m\_act++; // Incrementa minutos actuales  if( m\_act == 60 ) // De 60 reinicia en 0  m\_act = 0;  break;  case 5: h\_alrm++; // Incrementa la hora de la alarma  if( h\_alrm == 13 ) // De 13 pasa a 1  h\_alrm = 1;  if( h\_alrm == 12 ) { // En 12 ajusta bandera AM\_F\_A  if( AM\_F\_A ) { // y salidas AM y PM  AM\_F\_A = 0;  PORTB = PORTB & 0b11111101;  PORTB = PORTB | 0b00000100;  } else {  AM\_F\_A = 1;  PORTB = PORTB & 0b11111011;  PORTB = PORTB | 0b00000010;  }  }  break;  case 6: m\_alrm++; // Incrementa minutos de la alarma  if( m\_alrm == 60 ) // De 60 reinicia en 0  m\_alrm = 0;  break;  }  }  ISR(INT1\_vect) { // Atiende al boton DOWN  switch( modo ) {  case 3: h\_act--; // Decrementa hora actual  if( h\_act == 0 ) // De 0 pasa a 12  h\_act = 12;  if( h\_act == 12 ) { // En 12 ajusta bandera AM\_F  if( AM\_F ) { // y salidas AM y PM  AM\_F = 0;  PORTB = PORTB & 0b11111101;  PORTB = PORTB | 0b00000100;  } else {  AM\_F = 1;  PORTB = PORTB & 0b11111011;  PORTB = PORTB | 0b00000010;  }  }  break;  case 4: m\_act--; // Decrementa minutos actuales  if( m\_act == -1 ) // De -1 reinicia en 59  m\_act = 59;  break;  case 5: h\_alrm--; // Decrementa la hora de la alarma  if( h\_alrm == 0 ) // De 0 pasa a 12  h\_alrm = 12;  if( h\_alrm == 12 ) { // En 12 ajusta bandera AM\_F\_A  if( AM\_F\_A ) { // y salidas AM y PM  AM\_F\_A = 0;  PORTB = PORTB & 0b11111101;  PORTB = PORTB | 0b00000100;  } else {  AM\_F\_A = 1;  PORTB = PORTB & 0b11111011;  PORTB = PORTB | 0b00000010;  }  }  break;  case 6: m\_alrm--; // Incrementa minutos de la alarma  if( m\_alrm == -1 ) // De 60 reinicia en 0  m\_alrm = 59;  break;  }  }  *unsigned* *char* modo\_presionado() {  if( !( PIND & 0x02 ) ) { // Si modo est� presionado  for(*int* i = 0; i < 10; i++) // Espera realizando llamadas a mostrar  mostrar();  if( !( PIND & 0x02 ) ) // Si a�n sigue presionado regresa  return 1; // verdadero  }  return 0; // Sino, regresa falso  }  *int* main() {     // Configuracion de entradas y salidas     DDRB = 0b00001111; // Salidas: AM, PM, Zumbador y 2 puntos     DDRC = 0xFF; // Salida a los 7 segmentos     DDRD = 0xF0; // Botones e interruptor y salidas a       //transistores     PORTD = 0x0F; // Pull-Up en las entradas       //Configuracion del temporizador 2     TCCR2 = 0x04; // Para que desborde cada medio segundo     TIMSK = 0x40; // Interrupcion por desbordamiento     ASSR = 0x08; // Se usa al oscilador externo     sei(); // Habilitador global de interrupciones       // Configura interrupciones externas     MCUCR = 0B00001010; // INT1/INT0 por flanco de bajada (sin     // habilitar)       // Estado inicial del sistema: variables, banderas y salidas     h\_act = 12; m\_act = 0; s\_act = 0; // La hora actual inicia con 12:00     h\_alrm = 12; m\_alrm = 0; // La hora de la alarma es 12:00     modo = 1; // Modo normal, muestra la hora     AM\_F = 1; AM\_F\_A = 1; // Antes del medio doa     ALARM = 0; CAMBIO = 0; // Alarma apagada     PORTB = 0x02; // Solo activa la salida AM     while(1) { // Lazo infinito        mostrar(); // Exhibe, segon el modo        // Modo es una variable global        if( ALARM == 1 && (PIND & 0x01) ) {           ALARM = 0; // Apaga la alarma si esto encendida e           PORTB = PORTB & 0b11110111;// inhabilitada        }        if( modo\_presionado() ) {           modo++;           if( modo == 3 )           GICR = 0B11000000; // Habilita INT1 e INT0           else if( modo == 5 ) {              if( AM\_F\_A == 1) {                 PORTB = PORTB & 0b11111011; // Ajusta AM y PM segon la                 PORTB = PORTB | 0b00000010;// bandera AM\_F\_A de la alarma                 } else {                 PORTB = PORTB & 0b11111101;                 PORTB = PORTB | 0b00000100;              }              } else if( modo == 7 ) {              modo = 1;              GICR = 0x00; // Inhabilita INT1 e INT0              if( AM\_F == 1) {                 PORTB = PORTB & 0b11111011; // Ajusta AM y PM segon la                 PORTB = PORTB | 0b00000010; // bandera AM\_F de la hora actual                 } else {                 PORTB = PORTB & 0b11111101;                 PORTB = PORTB | 0b00000100;              }           }        } // Cierre del if de modo presionado     } // Cierre del lazo infinito  } // Fin de la funcion principal |

Antes de programar su funcionamiento, primero se realizó un bosquejo de cómo quedaría el diagrama para que en base a él se construyera y fuera más fácil su ensamblaje.

