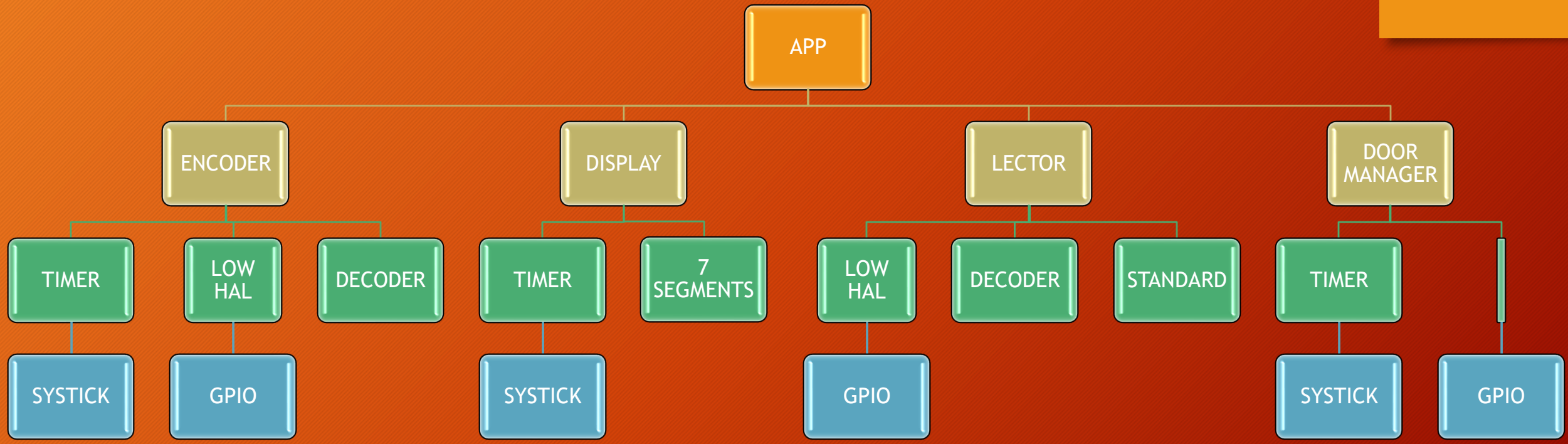
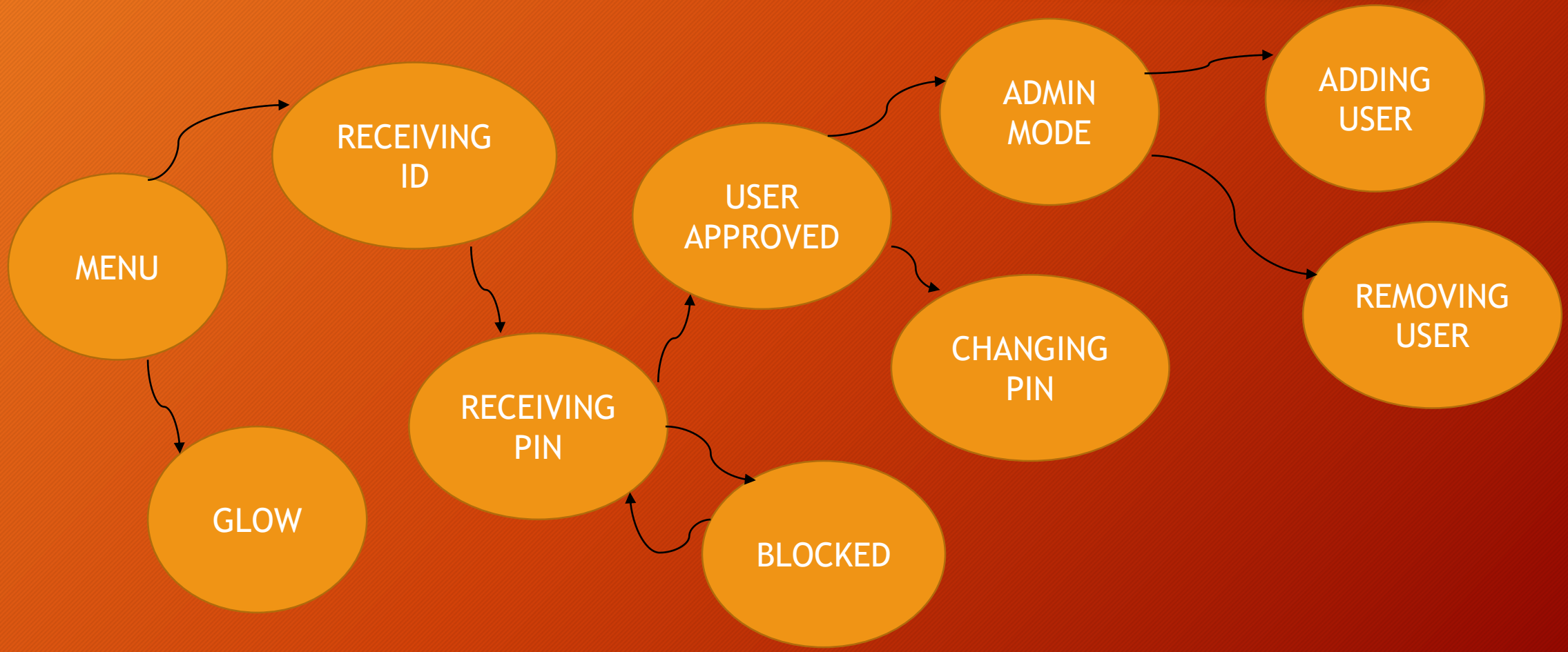
Control de Acceso

Laboratorio de Microprocesadores – ITBA

Grupo 5 :
Fernandez, Lucero - 57485
Mollón, Manuel - 58023
Vijande, Ezequiel - 58057
Larroque, Matías - 56597


APP
ENCODER
DISPLAY
LECTOR
DOOR MANAGER
TIMER
LOW HAL
DECODER
TIMER
7 SEGMENTS
LOW HAL
DECODER
STANDARD
TIMER
SYSTICK
GPIO
SYSTICK
GPIO
SYSTICK
GPIO

Aplicación


MENU
RECEIVING ID
GLOW
RECEIVING PIN
BLOCKED
USER APPROVED
ADMIN MODE
ADDING USER
REMOVING USER
CHANGING PIN

“Main program” - Aplicación

```
void App_Run (void)
{
    nameEvent = getEvent(&userData); // get new event
    switch(nameEvent){ // which type of event?
        case INPUT_EV:
            RestartTimer(INACTIVITY);
            nextState = (fsm.currentState.routines[INPUT_EV])(&userData); // action routine
            if(nextState.name != STAY){ // if state changes
                changingState = true;
            }
            break;
        case TIMER_EV:
            nextState = (fsm.currentState.routines[TIMER_EV])(&userData); // action routine
            if(nextState.name != STAY){ // if state changes
                changingState = true;
            }
            break;
        case KEYCARD_EV:
            RestartTimer(INACTIVITY);
            nextState = (fsm.currentState.routines[KEYCARD_EV])(&userData); // action routine
            if(nextState.name != STAY){ // if state changes
                changingState = true;
            }
            break;
        default:
            break;
    }
    if(changingState){
        fsm.currentState = nextState;
        changingState = false;
    }
}
```

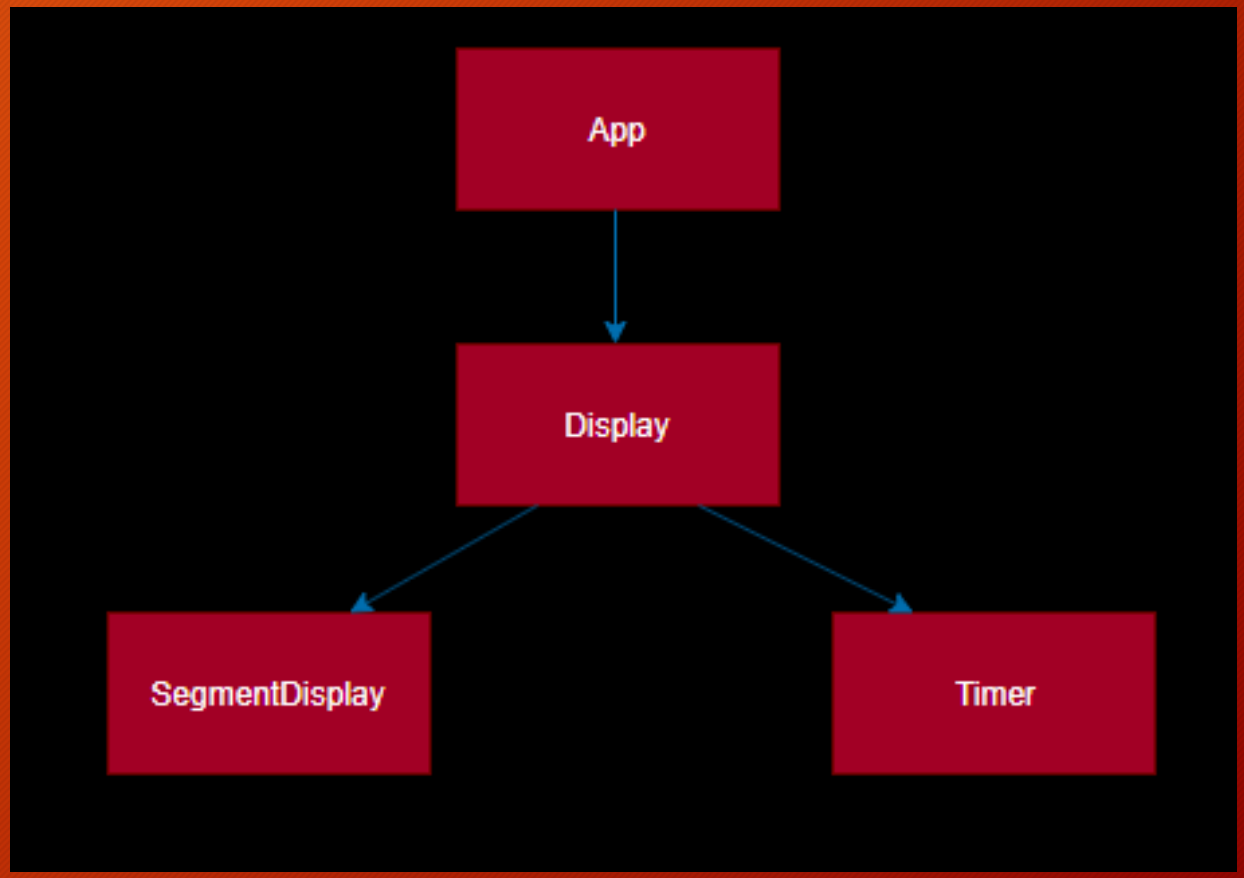
Display

Para la realización del display se utilizaron 3 módulos.

- El módulo display se ocupa de imprimir mensajes sin importar el hardware utilizado.

- El módulo SegmentDisplay funciona como driver que maneja el display de 7 segmentos de la placa utilizada.

- El módulo de Timer se utiliza para setear timers que indican el refresco de la imagen del display, el brillo y el movimiento del mensaje.



Display

```
//Inicializa los recursos necesarios para utilizar el display
void InitializeDisplay(void);

//Borra el contenido del display
void ClearDisplay(void);

//Imprime el string que recibe en el display.El segundo argumento indica si solo se desean mostrar
//los ultimos simbolos del string tal que el mensaje entre en el display(moving_string=false),
//o si se desea que el mensaje vaya apareciendo en el display de derecha a izquierda(moving_string=true).
void PrintMessage(const char* string, bool moving_string);

//Cambia la luminosidad del display, recibe un numero de 1 a 4,
//donde 1 simboliza 25% de intensidad y 4 simboliza 100%.
void SetBrightness(unsigned char brightness_factor);

//Actualiza el contenido del display.
void UpdateDisplay(void);
```

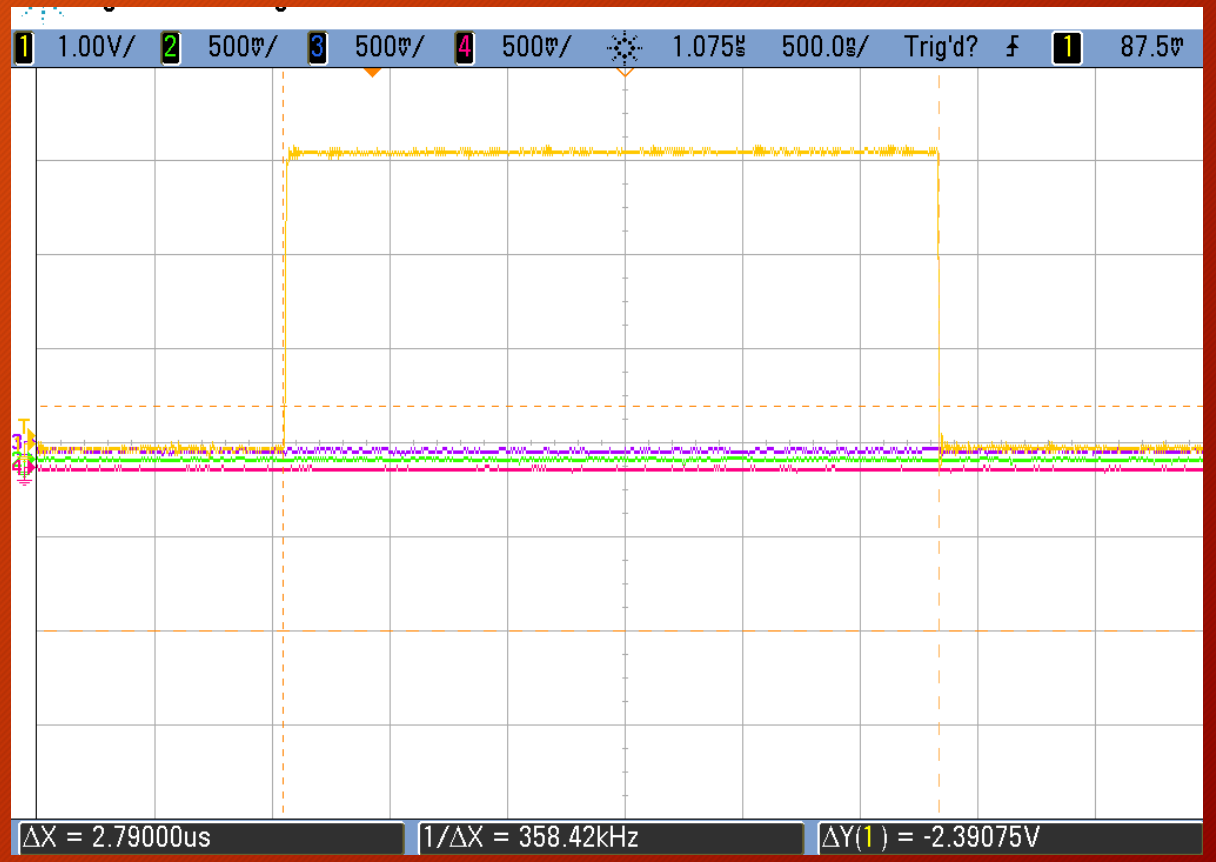
SegmentDisplay

```
void InitializeSegmentDisplay(void);
//Imprime el caracter c en la posicion que recibe.
void PrintChar(const char c,unsigned int pos);
```

- Es particular del hardware.
- Maneja pins GPIO del MicroControlador para actualizar la línea de selección del decoder y el valor de cada segmento.
- Guarda dos arreglos en memoria con los valores hexadecimales correspondientes a cada dígito.

Interrupción periódica

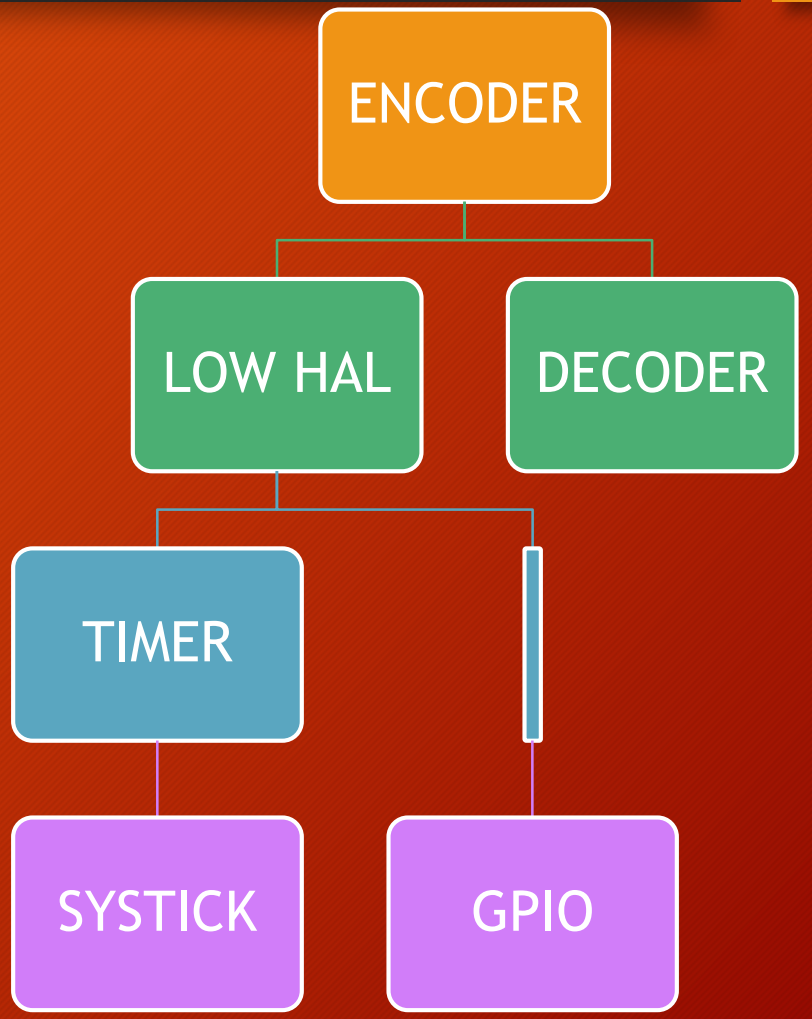
- Se ocupa de actualizar los timers y atender sus callbacks correspondientes.
- Genera los eventos correspondientes al display, los timers y el encoder.
- Dura alrededor de 3us y se llama cada 1ms,se emplea mediante Systick.
- Tiene un DutyCycle medido menor al 1% (alrededor de 0.3%)



Encoder

La realización del encoder se divide en 3 módulos.

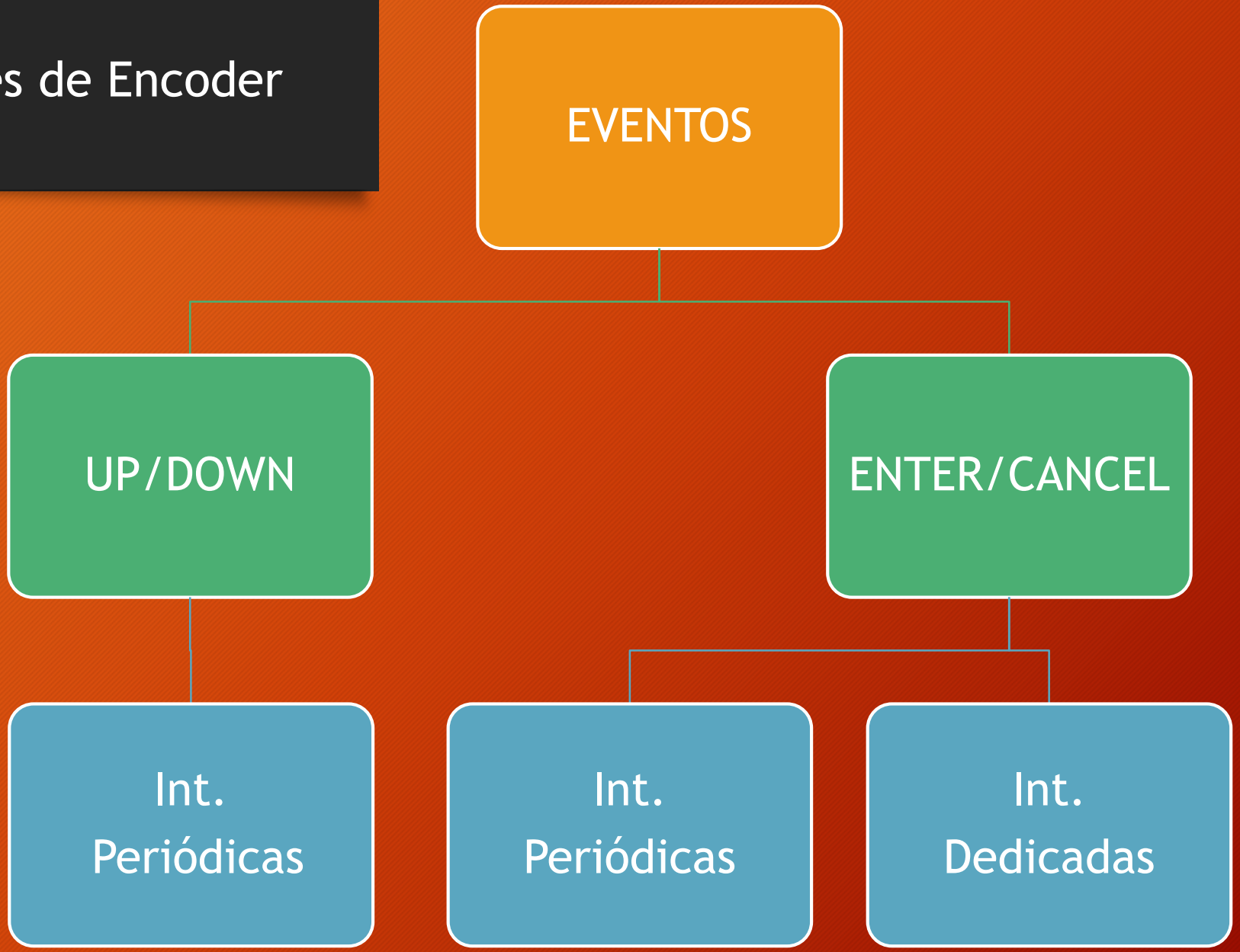
- El módulo **encoder** se ocupa de generar eventos abstrayéndose del hardware utilizado.
- El módulo **encoderDecoder** ayuda con la lógica para determinar qué ingresó el usuario mediante el encoder.
- El módulo **encoderHAL** se comunica tanto con timer.h como con gpio.h para la lectura de pines y seteo de callbacks de las interrupciones periódicas.



Eventos de Encoder

```
/*****************************************************************************
 *                                  DEFINICIONES
 *****************************************************************************/

typedef enum {UP/*horario*/, DOWN/*antihorario*/, ENTER, CANCEL, NO_ENC_EVENT}enc_type;

typedef struct{
    enc_type input;
    _Bool isValid;
}encoderUd_t;

/*****************************************************************************
 *                             FUNCIONES DEL HEADER
 *****************************************************************************/
//inicializa lo necesario para utilizar el encoder
void initializeEncoder(void);

//popea el primer evento de la cola
encoderUd_t popEncoderEvent(void);

//indica si el primer evento en la cola es válido
_Bool isEncEventValid(void);
```

Interrupciones de Encoder



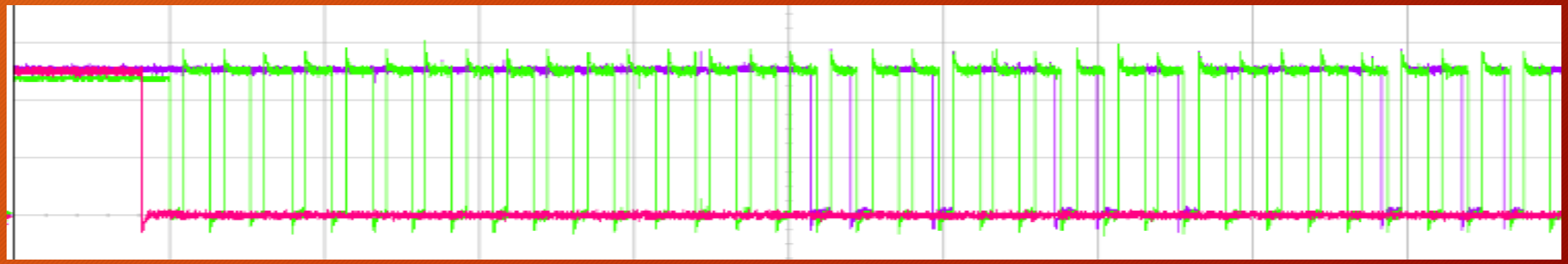
Lector de banda magnética

Inputs del MCU:

Enable: Señal que indica en que momento se desliza una tarjeta por el lector

Clock: Señal periódica de frecuencia proporcional a la velocidad de la tarjeta

Data: Señal que representa un dato (1 o 0) ante flanco descendente de clock



Lector de banda magnética

Interrupciones del MCU:

Enable – flanco ascendente: Se habilita clock, se comienza lectura.
(2 useg de duración)

Clock – flanco descendente: Se lee pin de datos y se guardan en un buffer.
(3 useg de duración)

Enable – flanco descendente: Se deshabilita clock, finaliza lectura, se genera un evento (lectureEvent).
(2 useg de duración)

Lectura de banda magnética

Manejo de evento de lectura:

Ante la generación de este tipo de evento, se guarda el mismo en una cola.

Una “APP” puede indicar que se extraiga un evento de la cola

Al extraer el evento de la cola (cuyo contenido es una serie de 1s o 0s), este es decodificado según los estándares especificados de tarjetas magnéticas

Finalmente, la APP tiene a su disposición una PALABRA decodificada (y separada en campos según el standard) , y la validez de esta palabra.

Lector de banda magnética

Decodificación:

Se averigua el tipo de TRACK que representan los datos de entrada.

Se descartan los datos correspondientes a bordes de la tarjeta.

Se decodifica cada carácter, separando en campos según los señaladores. A su vez, se validan los datos contemplando bits de paridad.

Door manager

Funciones disponibles para una “APP”:

openDoor() : prende led.

closeDoor() : apaga led.

openDoorTemporally() : prende led, y lo apaga luego de un tiempo configurable.

GRACIAS POR SU ATENCIÓN