|  |
| --- |
| Universidad Autonoma de Guadalajara |
| ETPS |
| Avances de Proyecto |
|  |
| **Issac Jimenez, Juan Mejía, Paul Molina** |
| **3/25/2017** |

|  |
| --- |
| Sistemas Embebidos en Tiempo Real |

# Nombre

ETPS

# Descripción General del Sistema

ETPS es un sistema capaz de brindar electroterapia gracias a un circuito de potencia formado por un específico hardware, el cual tiene de salida unos electrodos para colocar al paciente, a su vez, puede estar monitoreando la presión del cuerpo por medio del dispositivo MAX30100 y estarla desplegando con un grado de fiabilidad muy alto.

En la parte de la interacción HMI, el sistema cuenta con un LCD de 16x2 caracteres con el cual informara al usuario (dependiendo del tipo de aplicación que este seleccionada) datos de un correcto funcionamiento.



Figura 1. Diagrama General.

# Requisitos Funcionales



Figura 2. Diagrama de componentes y operaciones.

# Requisitos no Funcionales

**Tiempo**

* El sistema de monitoreo deberá cumplir con período máximo de 10ms
* La información en el LCD se actualizará con período máximo de 100ms
* El sistema heart beat deberá cumplir con un período máximo de 500ms, y finalizara su ejecución cuando el sistema se encuentre apagado.

**Fiabilidad**

* El sistema deberá brindar electro estimulación, sin importar que el sistema se encuentre monitoreando la presión en ese momento.
* El sistema deberá medir la presión del paciente de una manera no invasiva.
* Las funciones de electro estimulación y monitoreo de presión son independientes y no deberán provocar interferencia, conflicto o alterar su desempeño.

**Tolerancia a fallos**

* En caso de emergencia el sistema tendrá un paro de emergencia, el cual detendrá las funciones de electro estimulación.
* En caso de que falle el sistema de monitoreo de presión, el sistema de electro estimulación seguirá operando.

**Restricciones**

* El sistema deberá ser capaz de mostrar solamente un menú a la vez
* Las funciones de monitoreo de presión y electro estimulación podrán habilitarse si el switch de encendido se encuentra en la posición de encendido.
* El sistema deberá apagarse por completo si el switch de encendido se encuentra en la posición de apagado.
* El sistema no tendrá una precisión similar a la de un dispositivo clínico.
* La función de electro estimulación solo se podrán configurar, posicionándose en el menú de electro estimulación.

# Método de Diseño

Se utilizó la metodología HRT – HOOD, la cual facilita el diseño de la arquitectura lógica, la cual presentamos a continuación:



Figura 3. Arquitectura lógica del sistema.



Figura 4. Arquitectura lógica del subsistema Drivers.

# Definición de HW y SW

* 1. Definición de HW

ETPS cuenta con un Hardware tanto adquirido como hecho por nosotros a partir de simple componentes:

* Microcontolador Kinetis KL25Z128 (NXP)
* Sensor MAX30100 para obtener la medición del pulso cardiaco
* LCD 16x2
* 5 Push Buttons
* Buzzer
* Circuito de potencia diseñado(Transistores, Diodos, Resistencias, Transformador)
* Electrodos
* LEDs (Rojo, Azul, Amarillo)  
  1. Definición de SW

La definición del software está completamente basada en los requisitos, haciendo a ETPS un sistema embebido de calidad en su respuesta y cumplimiento de tareas. Se hicieron capas 4 capas importantes en la definición del software:

* App
* MidDrivers
* LowDrivers
* System

Se realizó un análisis específico para cada necesidad del sistema y se fue agrupando por funcionalidad y características, al tener todos los componentes de software que cumplen con los requisitos, se agruparon en estas 4 capas que a su vez, cada una tiene distintos componentes

* App: (Electroterapia, HMI, Presión, Alertas, HB)
* MidDrivers: (DBNCR, IRQ, LCD)
* LowDrivers: (GPIO, IIC, PIT, TPM)
* System(Scheduler, SystemInit)

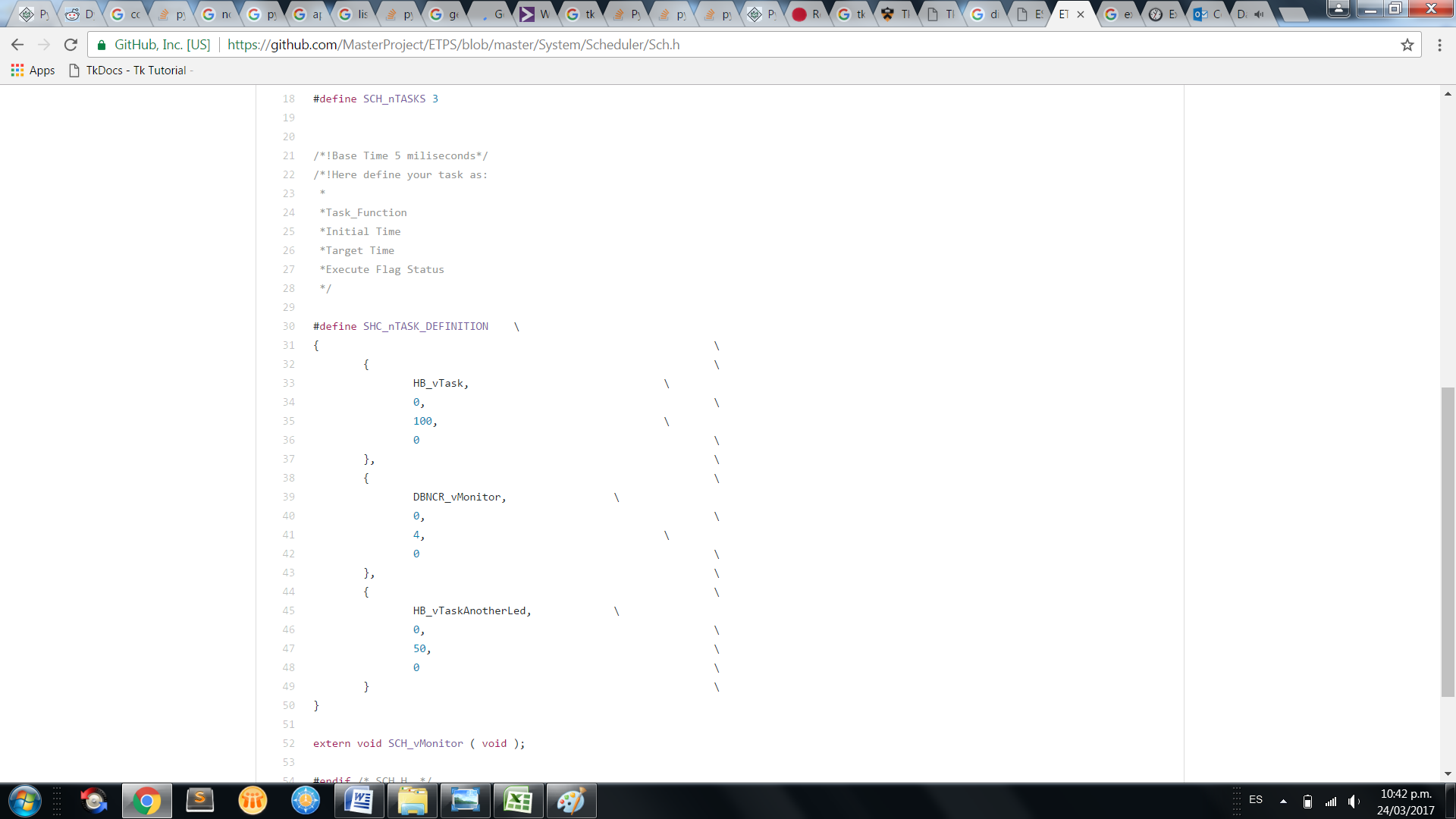
# Análisis de Planificación

Las tareas se listan a continuación:

* Electrotherapy, (50 ms)
* Warnings(10 ms)
* HMI(100 ms)
* PressureSensor(10 ms)
* LCDDriver(5 ms)
* HeartBit(500 ms)
* Debouncer(20 ms)

No se utilizó un sistema operativo, por lo cual se diseñó un método para que las tareas se compartan el uso del procesador, la técnica utilizada es con una interrupción que se ejecuta cada 5 ms. Cada tarea tiene un contador el cual se va comparando (contra un tiempo target) y aumentándose (en el caso de no llegar al target) dentro de la interrupción, una vez que se alcanza el tiempo target se levanta una bandera, la cual da el permiso de que una tarea puede ser ejecutada.







|  |  |  |  |
| --- | --- | --- | --- |
| Tarea | Tiempo de Computo | Periodo | U |
| LCD Driver | 1 ms | 5 ms | .2 |
| Warnings | 1 ms | 10 ms | .1 |
| Pressure | 1 ms | 10 ms | .1 |
| Debouncer | 1 ms | 20 ms | .05 |
| ElectroTherapy | 1 ms | 50 ms | .02 |
| HMI | 1 ms | 100 ms | .01 |
| HeartBIt | 1 ms | 500 ms | .002 |
| Total U | | | .482 |



8-. Diseño de Software y Hardware.

Software

Dentro de la parte de software se dividio en 4 bloques principales: Low Drivers, Mid Drivers, HMI y App, los cuales contienen modulos dentro de ellos. A continuación se muestran los modulos de cada bloque.

Low Drivers

* ADC

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **ADC\_vInit** | - | - | Inicializa el ADC del Microcontrolador |
| **ADC\_u16ReadChannel** | uint8 u8Channel | uint16 | Retorna el valor del ADC seleccionado |

* GPIO

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **GPIO\_vPinsInit** | - | - | Inicializa los puertos del Microcontrolador |

* IIC

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **Pause** | Int number | - | Genera una pausa |
| **IIC\_vInit** | - | - | Inicializa el IIC |
| **IIC\_vWrite** | uint8 u8SlaveAddress, uint8 u8RegisterAddress, uint8 u8Data | - | Escribe un dato mediante el protocolo IIC |
| **IIC\_u8Read** | uint8 u8SlaveAddress, uint8 u8RegisterAddress | uint8 | Retorna el valor leído por el protocolo IIC |

* PIT

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **PIT\_vInit** | - | - | Inicializa el PIT (programmable interrupt timmer) |

* TPM

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **TPM\_vInit** | - | - | Inicializa el módulo TPM del microcontrolador |
| **TPM\_vSetPWM** | TPM\_tenModules enModule, TPM\_tenChannels enChannel, uint16 u16Value | - | Pone el valor deseado de PWM al canal requerido |
| **TPM\_vChangeDutyCycle** | TPM\_tenModules enModule, TPM\_tenChannels enChannel, uint8 u8DutyCycle | - | Cambia el ciclo de trabajo del canal requerido |
| **TPM\_vChangeFrecuency** | TPM\_tenModules enModule, TPM\_tenChannels enChannel, uint16 u8Frecuency | - | Cambia la frecuencia del módulo del TPM |

Mid Drivers

* BUZZER

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **BZR\_vMain** | - | - | Monitorea el estado del buzzer y envíao deja de enviar la señal para hacerlo sonar |
| **BZR\_vActivateBuzzer** | - | - | Cambia el estado del buzzer a activo |
| **BZR\_vDeactivateBuzzer** | - | - | Cambia el estado del buzzer a inactivo |

* DBNCR

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **DBNCR\_vUpdate** | - | - | Actualiza el estado del botón cuando es presionado y le aplica un “debounce” para asegurarse de una lectura correcta |
| **DBNCR\_vMonitor** | - | - | Monitorea cada cierta cantidad de tiempo, definida en el scheduler, si un botón es presionado |
| **DBNCR\_u8ReadButtonState** | DBNCR\_tenButtons enButton | uint8 | Retorna si el botón requerido ha sido leido |

* IRQ

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **IRQ\_vInit** | - | - | Inicializa las interrupciones del Microcontrolador |
| **PORTA\_IRQHandler** | - | - | Maneja las interrupciones del puerto A |

* LCD

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **LCD\_vInit** | - | - | Inicializa la LCD |
| **LCD\_vDriver** | - | - |  |
| **\_vIdle** | - | - | No hace nada la LCD |
| **\_vWriteCMD** | - | - | Manda instrucciones de comando a la LCD |
| **\_vWriteData** | - | - | Manda datos para que se muestren en la LCD |
| **\_vEnable** | - | - | Enciende la LCD |
| **\_vDisable** | - | - | Apaga la LCD |
| **\_vDelayCMD** | - | - | Retardo necesario para que la LCD lea cada comando |
| **LCD\_vUpdateScreen** | - | - | Actualiza la LCD cada determinado tiempo |
| **LCD\_vClearScreen** | - | - | Limpia la LCD |
| **LCD\_u8IsBusy** | - | uint8 | Indica cuando la LCD está siendo usada por otro proceso |

HMI

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **\_vWelcomeScreen** | - | - | Muestra la pantalla Inicial |
| **\_vTherapySelectedScreen** | - | - | Muestra la pantalla de Electroterapia |
| **\_vPressureSelectedScreen** | - | - | Muestra la pantalla de medición de presión |
| **\_vFreqSelectedScreen** | - | - | Muestra la pantalla de selección de frecuencia |
| **\_vIntsSelectedScreen** | - | - | Muestra la pantalla de selección de intensidad |
| **\_vFreqValueScreen** | - | - | Actualiza el valor de frecuencia en la pantalla |
| **\_vIntsValueScreen** | - | - | Actualiza el valor de intensidad en la pantalla |
| **\_vPressureValueScreen** | - | - | Actualiza el dato de la medición de presion |
| **\_vClearScreen** | - | - | Borra el contenido de la pantalla |
| **\_vEmergencyState** | - | - | Muestra la pantalla de paro de emergencia |
| **\_vPswrdState0** | - | - | Muestra la pantalla para introducir la contraseña que desactiva el paro de emergencia |
| **\_vPswrdState1** | - | - | Muestra la pantalla de contraseña con un carácter escrito |
| **\_vPswrdState2** | - | - | Muestra la pantalla de contraseña con dos caracteres escrito |
| **\_vPswrdState3** | - | - | Muestra la pantalla de contraseña con tres caracteres escrito |
| **\_vPswrdState4** | - | - | Muetra la pantalla de contraseña incorrecta |
| **HMI\_vMain** | - | - |  |
| **HMI\_vSetEmergencyStop** | - | - |  |
| **HMI\_vGetEmergencyStop** | - | - |  |
| **HMI\_u8GetFreqValue** | - | - |  |
| **HMI\_u8GetIntsValue** | - | - |  |

APP

* ElectroTherapy

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **\_ET\_vMain** | - | - | Actualiza la frecuencia de la Electroterapia haciendo la conversión entre lo que despliega la LCD y la instrucción que se le envía al microcontrolador |

* HeartBeat

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **\_HB\_vInit** | - | - | Inicializa el LED de Beart Beat |
| **\_HB\_vTask** | - | - | Intercala entre ambos estados del LED |
| **HB\_vActiveYellowLed** |  |  | Cambia el estado del LED a encendido |
| **\_HB\_vDeactiveYellowLed** | - | - | Cambia el estado del LED a apagado |

* Frequency (nombrado como test)

|  |  |  |  |
| --- | --- | --- | --- |
| Función | Parámetros de entrada | Valor de retorno | Descripción |
| **TEST\_vChangingFreq** | uint8 u8Freq | - | Actualiza los registros del micro para variar la frecuencia introducida |