ECE 411

11/8/17

Team 15:

Zach Stamler

Linyi Hong

Nathaniel Dusciuc

Andrew Capatina

Detailed Design:

## Speed Demon Speedometer



**Level 0**

|  |  |
| --- | --- |
| Module | Speedometer |
| Inputs | * Sensor * Power supply: various DC power * 5- way tactile switch |
| Outputs | * LCD screen |
| Functionality | Receive signal from sensor,calculate/display speed |



## Level 1



|  |  |
| --- | --- |
| Module | Sensor Module |
| Inputs | * Proximity-induced magnetic field flux from wheel magnet * +~4VDC (direct from battery) |
| outputs | * Binary pulses |
| Functionality | Provide edge-detectable information about wheel location. |

## Level 1



|  |  |
| --- | --- |
| Module | External Crystal Oscillator |
| Inputs | * Thermal noise/OSC loop |
| outputs | * 32.768kHz Clock |
| Functionality | Provide a stable clock reference for the MCU |

## Level 1



|  |  |
| --- | --- |
| Module | 5-Way Switch |
| Inputs | * User tactile input |
| outputs | Binary ‘button press’ data. NP = +3.3VDC, P = GND   * CENTER * UP * DOWN * LEFT * RIGHT |
| Functionality | Provide user interfacing and device control capability |

## Level 1



|  |  |
| --- | --- |
| Module | 8x2 LCD display |
| Inputs | * 8-Wire binary data bus from MCU + 3 ctrl inputs (tot. 11) * 3.3V DC Power supply |
| outputs | [Visual information] |
| Functionality | Visualize the speed by showing it on the screen |

## Level 1



|  |  |
| --- | --- |
| Module | Power supply |
| Inputs | * ON/OFF switch (CENTER) * 5V from USB |
| Outputs | * +3.3VDC regulated power * Raw battery power (~+4VDC) |
| Functionality | Power supply for the device |



**Level 2**

|  |  |
| --- | --- |
| **Master Module** | **Power Supply** |
| Submodule | USB Power Inlet |
| Inputs | * USB connection (all pins except PWR/GND are NC) |
| outputs | * ~+5VDC |
| Functionality | Source of charging power for battery |

|  |  |
| --- | --- |
| **Master Module** | **Power Supply** |
| Submodule | Li-Po battery |
| Inputs | * Current from charge controller |
| outputs | * ~+4VDC |
| Functionality | Power source for device |

|  |  |
| --- | --- |
| **Master Module** | **Power Supply** |
| Submodule | Charge controller |
| Inputs | * USB power * Battery power |
| outputs | * Battery charging current * Battery power thru |
| Functionality | Routes power either from USB inlet to battery, or from battery to rest of device |

|  |  |
| --- | --- |
| **Master Module** | **Power Supply** |
| Submodule | ON/OFF Switch Timer Circuit |
| Inputs | * Center switch push * Raw battery power |
| outputs | * Binary ON/OFF signal @ batt voltage |
| Functionality | User control for device power state. Outputs ‘on’ (batt voltage) with a momentary press of center button, outputs ‘off’ (GND) after long press of ~1s. |

|  |  |
| --- | --- |
| **Master Module** | **Power Supply** |
| Submodule | Power OP Defeat Logic |
| Inputs | * ON/OFF signal from timer switch circuit * USB power (+5VDC) |
| outputs | * Binary EN signal |
| Functionality | Controls device power state using EN input of DC-to-DC converter. Follows signal from ON/OFF Timer Circuit unless USB power plugged in. Disables power when USB present. Circuit is powered by ON/OFF signal from timer circuit. |

|  |  |
| --- | --- |
| **Master Module** | **Power Supply** |
| Submodule | DC-to-DC Converter |
| Inputs | * Binary EN signal from Power OP Defeat Logic * DC power from charge controller |
| outputs | * +3.3V regulated power |
| Functionality | Switching power regulation circuit. Input power can be either battery or USB power, but EN should be off if USB power applied, so functional input should always be battery voltage. |

## Level 1



|  |  |
| --- | --- |
| Module | Microcontroller |
| Inputs | * 1 bit Sensor inputs * 3.3V DC power supply * CENTER: Power on and select * LEFT: Left operation * RIGHT: Right operation * UP: Up operation * DOWN: Down operation |
| Outputs | * Data to LCD |
| Functionality | * Receive signal from input, calculate the speed, output to LCD |



Software:

|  |  |
| --- | --- |
| Module Name | main() |
| Module Type | Coordination Module |
| Input Arguments | * none |
| Output Arguments | * Return 1 when operations are done. |
| Description | Main function begins by initializing the LCD with LCD\_initialize() function call.  speed\_conversion() shall convert the input argument count to speed. Upon receiving the returned speed, double\_to\_str() is called to convert the value to a string. LCD\_display() shall output the string argument to the LCD. The user\_settings() function will also be called to allow the user to change the device's settings. The user interaction will involve a button switch to switch between the operating modes. |
| Modules Invoked | speed\_conversion, LCD\_display, user\_settings(), double\_to\_str(), LCD\_initialize. |

|  |  |
| --- | --- |
| Module Name | LCD\_Initialize() |
| Module Type | Coordination Module |
| Input Arguments | * none |
| Output Arguments | * Returns 1 |
| Description | Function will make a series of call to LCD\_sendCommand() to set up the screen and cursor of the LCD. |
| Modules Invoked | LCD\_sendCommand(). |

|  |  |
| --- | --- |
| Module Name | speed\_conversion() |
| Module Type | Transform module. |
| Input Arguments | * count: integer to store period between each sensor pulse. * wheel\_size |
| Output Arguments | * speed: double to store calculated speed. |
| Description | The parameter ‘count’ and ‘wheel\_size’ is passed in and the appropriate operations are performed to convert the count to speed. ‘speed’ is then returned to main. |
| Modules Invoked | none |

|  |  |
| --- | --- |
| Module Name | double\_to\_str |
| Module Type | Transform module. |
| Input Arguments | * speed: double which represents value of speed. |
| Output Arguments | * speed\_str: string representation of value of speed. |
| Description | This function will be invoked by main to convert the parameter ‘speed’ to a string. |
| Modules Invoked | none |

|  |  |
| --- | --- |
| Module Name | LCD\_sendString() |
| Module Type | Input Module |
| Input Arguments | * stringVal: String to be sent to LCD |
| Output Arguments | * Returns 1 |
| Description | Function will select the LCD that will be written to, set LCD to character mode, enable the LCD, and set the parameter ‘stringVal’ to be displayed to the LCD. All signals will be sent to the GPIO output pins. |
| Modules Invoked | none |

|  |  |
| --- | --- |
| Module Name | LCD\_sendCommand() |
| Module Type | Input Module |
| Input Arguments | * stringVal: String to be sent to LCD |
| Output Arguments | * Returns 1 |
| Description | Function will select the LCD that will be written to, set LCD to command mode, enable the LCD, and write ‘stringVal’ out to the LCD. |
| Modules Invoked | none |

|  |  |
| --- | --- |
| Module Name | user\_settings() |
| Module Type | Transform Module |
| Input Arguments | * Float current\_wheel\_size |
| Output Arguments | * Returns floating integer wheel\_size. |
| Description | Function will call LCD\_sendString() to display options to the user. User interaction will involve button presses to overwrite the current value of  the wheel size. Function will then concatenate result and return ‘wheel\_size’ main. |
| Modules Invoked | none |