# **Group 5 Library 1 Components**

Authors: Nash Martin and Vinessa Almanza

**Definitions**

void E4235\_Select(int GPIO\_Number, int Value)

* **Functionality:** Select is used to set a GPIO pin as an input or output. This is to be used before using read() or write(). Additionally, if an invalid input is given for either parameter a corresponding error message will be printed to the console. This only needs to be called once to set a single GPIO as an input or output for a program.
* **Parameters:**
  + GPIO\_Number: is a valid GPIO value from 0 through 29. Note that this is NOT the pin value of a GPIO. If an invalid value is provided (i.e. outside of 0 - 29, an error message will be printed).
    - E.g. GPIO3 means GPIO\_Number = 3
    - E.g. GPIO\_Number = -1 results in the following error "GPIO number not valid, please provide a valid GPIO number.”
  + Value: to set the GPIO as an input, use 0, otherwise use 1 to set as an output. Note that if an invalid value is given, an error message will be displayed.
    - E.g. Value = 0, sets GPIOXX to input, where XX is a valid GPIO value
    - E.g. Value = 1, sets GPIOXX to output
    - E.g. Value = 2, results in the following error “Value not valid, please provide a valid value”

void E4235\_Write(int GPIO\_Number, int Value )

* **Functionality:** Write is used to write a HIGH/LOW value to a given GPIO pin. It will stay HIGH/LOW unless overridden by another write invocation or if the pi is turned off. Note that write DOES NOT contain the backend to set up the RP4 pins therefore will require the invocation of select() beforehand. Use the select function to set the desired GPIO as an output. The function returns void but will output an error message if invalid values for the parameters were entered.
* **Parameters:** 
  + GPIO\_Number: is a valid GPIO value from 0 through 29. Note that this is NOT the pin value of a GPIO.
    - E.g. GPIO3 means GPIO\_Number = 3
  + Value: represents outputting a HIGH (represented by 1) or a LOW (represented by 0) to the specified GPIO.
    - E.g. To output a HIGH to GPIO3, set Value = 1.

int E4235\_Read(int GPIO\_Number)

* **Functionality:** Read is used to read the current value of a specified GPIO pin. Note that only valid values of the GPIO number can be read, else an error message is outputted. This function does not require a setup/init function to be invoked beforehand and can be called on its own (however it is suggested if more functionality is desired). The function returns an integer and will return a -1 if an error occurred. The user can add their own error message in C by making a conditional statement for if the read() returns -1.
* **Parameters:**
  + GPIO\_Number: is a valid GPIO value from 0 through 29. Note that this is NOT the pin value of a GPIO.
* **Outputs:**
  + Returns an int of 0 or 1 which refers to a LOW or HIGH detected at the specified GPIO pin respectively. Else, returns -1 to indicate an error occurred i.e. invalid number is entered. Additionally, when an invalid GPIO value is entered the following error message will be printed "GPIO number not valid, please provide a valid GPIO number.”.
  + Additionally, if you use the ASM call to read(), utilize a system call to printf to print out the value returned by read in R0.

\*Note: refer to the examples on the next page for how to call read() and write().

**ASM Calling select(), write(), and read()**

|  |
| --- |
| **@invoking select, read, and write functions in ASM**  **.global main**  **.extern E4235\_Select**  **.extern E4235\_Write**  **.extern E4235\_Read**    **.text**  **Main:**  **mov r0, #21 @selecting GPIO 21**  **mov r1, #1 @setting GPIO 21 as an output**  **bl E4235\_Select @invokes the select function**  **mov r0,#21 @using GPIO 21, sets r0 = 21**  **mov r1, #1 @setting GPIO 21 to HIGH, sets r1 = 1**  **bl E4235\_Write @invokes write function**  **mov r0,#21 @using GPIO21, sets r0 = 21**  **bl E4235\_Read @calling read function**  **mov r7,#1 @end program, note: the value returned by read() is stored in R0**  **svc 0** |

**C Calling select(), write(), and read()**

|  |
| --- |
| **#include <stdio.h>**  **#include <stdlib.h>**  **Extern void E4235\_Select(int, int);**  **extern void E4235\_Write(int, int);**  **extern int E4235\_Read(int);**  **int main(){**    **// calling select to set GPIO 21 as an output**  **E4235\_Select(21,1);**  **// Testing the write function**  **E4235\_Write(21,0);**  **int val = -1;**    **// Testing the read function**  **val = E4235\_Read(31);**    **if (val == -1){**  **printf("Invalid GPIO Number\n");**  **}**  **else{**  **printf("GPIO LEVEL = %d\n", val);**  **}**  **return 0;**    **} // main** |

## **References**

[1] Used for read implementation: <https://forums.raspberrypi.com/viewtopic.php?t=49444>

[2] GPIO functionality: <https://datasheets.raspberrypi.com/bcm2711/bcm2711-peripherals.pdf>

[3] Used for write implementation: <https://gist.github.com/mathis-m/facd241fe1f324c7b22338484f60338f>

[4] How to use mmap: <https://bob.cs.sonoma.edu/IntroCompOrg-RPi/sec-gpio-mem.html>