

**ECSE 4235: Embedded Systems II**

Library 2

Alex George, Christopher Ghattas

Group 3 // 04/15/2024

e4235\_MultiGPIOWrite.s

The `e4235\_MultiGPIOWrite` function is responsible for managing GPIO operations, both single and multiple, within the assembly code. It accepts various parameters stored in registers, including the GPIO value to be written (R0), the writing mode (R1), and the address indicating the GPIO (R2). Additionally, it utilizes registers R3 to R10 for extracting digits and determining the end of the input strings. The function begins by initializing the stack and setting up memory locations for stack pointers. It then proceeds with different logic paths based on the number of digits in the GPIO, handling cases for single and double-digit GPIO values separately.

For single-digit GPIO values, it directly processes the value and writes it to the GPIO pin. For double-digit GPIO values, it converts the ASCII to numerical values and combines them before performing the GPIO operation. The function also includes logic for reading and writing to single GPIO pins, as well as handling multiple GPIO operations.

Then, the function iterates through the input string, processing each character to determine GPIO values and actions to perform. It ensures proper handling of null characters to mark the end of the string. When finished, the function restores the stack and returns, ensuring proper memory management. The data section includes strings representing GPIO values and memory locations for stack pointers, essential for maintaining state and executing operations effectively. A chart with all variables used is shown below.

**Parameters**

| R0 | Holds the current GPIO value to be written. |
| --- | --- |
| R1 | Static. Can hold 0 or 1 for writing high or low. |
| R2 | Static. Holds the address passed from R0. |
| R3 | Holds the first digit of the GPIO. |
| R4 | Holds the second digit of the GPIO. |
| R5 | Holds the current GPIO when writing to multiple GPIOs. |
| R6 | Holds the maximum GPIO value when writing to multiple GPIOs. |
| R7 | Indicates whether a null character has been encountered, marking the end of the string. |
| R8 | Holds the counter of how many digits are in the GPIO number. |
| R9 | Spare register, holds LR address in some cases. |
| R10 | Holds the third digit if needed. |

**Functionality**

* Stack Setup: Saves the LR register and registers R0-R12 onto the stack.
* SP\_BLOCKTWO Setup: Loads the stack pointer into a memory location.
* Initialization: Initializes registers R2, R8, and R7.
* Single Digit GPIO: If the GPIO only uses one digit, it subtracts 48 from the digit (ASCII adjustment) and moves it to R0.
* Two Digit GPIO: If the GPIO uses two digits, it subtracts 48 from each digit, converts them to their corresponding numerical values, and combines them into R0.
* Single GPIO Write: Writes the value of a single GPIO pin indicated by R0.
* Single GPIO Read: Reads the value of a single GPIO pin indicated by R0.
* Multiple GPIO Write: Writes values to multiple GPIO pins, starting from the current GPIO in R6 and ending at the maximum GPIO in R6. It iterates through each GPIO, writing the value to each one.
* Multiple GPIO Read: Reads values from multiple GPIO pins, starting from the current GPIO in R6 and ending at the maximum GPIO in R6. It iterates through each GPIO, reading the value and accumulating it in R7.
* Dash Case: Handles cases where GPIO values are separated by a dash ('-'). It extracts the GPIO values, determines the range, and writes to each GPIO within the range.
* Read or Write Determination: Determines whether to read or write to the GPIO based on the number of digits in the GPIO value.
* Center Logic Loop: Iterates through the input string, processing each character to determine the GPIO values and actions to perform.
* End: Restores the stack pointer and registers and returns.

**Data Section**

| S1 | String representing GPIO values. |
| --- | --- |
| output | Format string for output. |
| SP\_BLOCK | Memory location for stack pointer. |
| SP\_BLOCKTWO | Memory location for stack pointer |

s1: String representing GPIO values.

output: Format string for output.

SP\_BLOCK: Memory location for stack pointer.

SP\_BLOCKTWO: Memory location for stack pointer

e4235\_MultiGPIORead.s

The `e4235\_MultiGPIORead` function reads values from GPIO pins, both individually and in groups. Similar to the previous function, it utilizes registers R0-R10 for various parameters and operations. On start, the function initializes the stack and memory locations for stack pointers, preparing for subsequent operations. It also initializes register R11 to store the output binary read result, and sets up other necessary registers for processing.

The function then proceeds with logic paths based on the number of digits in the GPIO, which differentiates between single and double-digit GPIO values. Single-digit GPIO values are processed straightforwardly, while double-digit values involve converting ASCII characters to numerical values and combining them appropriately, similar to the last functoin. Following this, the function includes distinct paths for reading from single GPIO pins and handling multiple GPIO operations, similar to the `e4235\_MultiGPIOWrite` function.

When finished, the function will restore the stack and return the read result stored in R11. Additionally, it includes a section in the data segment for storing a character array and defining memory locations for stack pointers.

**Parameters**

| R0 | Holds the current GPIO value to be read. |
| --- | --- |
| R1 | Static; indicates the writing mode (0 or 1 for write high or low). |
| R2 | Static; holds the address passed from R0. |
| R3 | Temporarily holds the first digit of the GPIO. |
| R4 | Temporarily holds the second digit of the GPIO. |
| R5 | Holds the current GPIO when reading from multiple GPIOs. |
| R6 | Holds the maximum GPIO value when reading from multiple GPIOs. |
| R7 | Signals the encounter of a null character, marking the end of the string. |
| R8 | Tracks the number of digits in the GPIO number. |
| R9 | Spare register; used for various operations |
| R10 | Holds the third digit if necessary. |
| R11 | Stores the output binary read result. |

**Functionality**

* Initializes the stack and memory locations for stack pointers.
* Handles single and double-digit GPIO values separately.
* Processes single and multiple GPIO read operations.
* Manages GPIO values presented as a range separated by a dash ('-').
* Iterates through the input string, processing each character to determine GPIO values.
* Properly handles null characters to mark the end of the string.
* Restores the stack and returns the read result stored in R11.

**Data Section**

| S1 | Stores a character array. |
| --- | --- |
| output | Defines a format string for output. |
| SP\_BLOCK | Memory location for stack pointer. |
| SP\_BLOCKTWO | Memory location for stack pointer. |

This documentation provides a structured overview of the `e4235\_MultiGPIORead` and ‘e4235\_MultiGPIOWrite’ functions, highlighting its parameters, functionality, and associated data segments for multi GPIO read/write operations in assembly code.