1A – Shaun Davis, Trinity Merrell, Logan Smith, and Khaled Alfayez

Professor Micah

CSSE232

11 January 2017

**The Creation of a Processor: Milestone 1**

## **Registers:**

There are 19 registers that we’ll use in our processor. Ten of which are general purpose registers, similar to $sn registers in MIPS and follows the same convention of backing up stored values before use. We also have an arithmetic result register dedicated for storing the result of any arithmetic instruction. We’ll also have a register for return addresses, similar to MIPS’s $ra register. Also, we’ll have 2 argument registers for jumping and linking between functions and one return register, similar to MIPS’s $v0 register. Included also is a display register, used to help display stuff on a 16 bit LCD screen. We’ll also have a cause register in another coprocessor for handling exceptions. Plus, we will have 2 OS registers, similar to $k*n* registers in MIPS.

To summarize, here is a table:

|  |  |  |
| --- | --- | --- |
| Registers | Availability | Description |
| $s*n* (0-9) | Read/write | These registers are general purpose registers that can be used by the user to store whatever they want. |
| $ar | Read | This register holds the result of any instruction that gives a result. Similar to the $at register in MIPS or an accumulator register, except that the user can copy/read the stored value. |
| $ra | Read/write | This register is basically the $ra register from MIPS. It is where the return address is stored. |
| $a*n* (0-1) | Read/write | These registers are used to store arguments for use in a called function. |
| $v | Read/write | This register is for storing the return value from a function. It is similar to MIPS’s $v registers. |
| $d | Read/write | This register is used for communication between the display and the processor. |
| $ex | Read | This is the cause register and will be on the coprocessor with its own instructions. |
| $k*n* (0-1) | Read/write  (while handling exceptions) | Similar to MIPS $k*n* registers, these 2 registers are dedicated for exception handling. |

# **Procedure Call Conventions:**

For meeting.