CST250 Project 3: UART Calculator

Learning Objectives:

- Create modular code in a collaborative environment

The Task

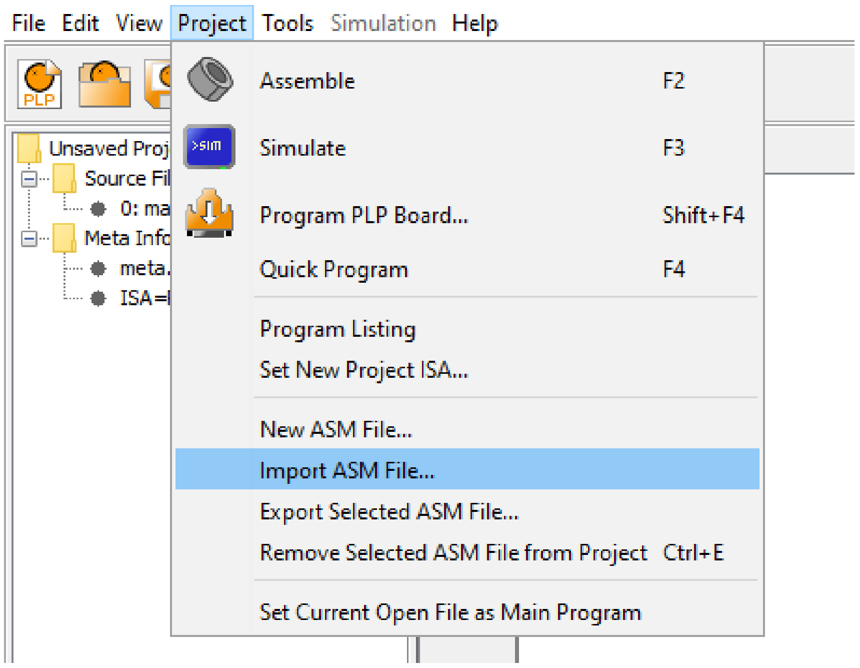
In this project you will be combining and building on what you have done in the first 2 projects. You are free to reuse any code that you wrote in the previous 2 projects or the provided Project 2 solution. This program should be a full calculator that uses the UART for both input and output.

Calculator Operation

You calculator should accept full mathematical expressions with 2 numbers and an operator (e.g. “1+2=”) and print the result to the UART using the provided module. It should handle the operators to add (+), subtract (-), and multiply (*), but it **does not** need to handle division. It should accept multiple expressions and generate the appropriate outputs for each expression without manually resetting the simulation (e.g. the input “5-4=1*2=” should output “1” followed by “2”).

UART Module

The UART Module is available on Blackboard as an *ASM* file. You can import it into a PLP project by navigating to “*Project*”, selecting “*Import ASM file*”, and then selecting the UART Module *ASM* file.



The module contains a print function, *project_3_output_number*, that uses 2 registers as input arguments, `$a0` and `$a1`. The register, `$a0`, is the value to be displayed on the UART and register `$a0` is used to indicate the error message. **If $a1 is non-zero an error message will be displayed** regardless of the value in `$a0`. The table below indicates the mapping between `$a1` and the UART output.

$a0	**$a1**	**UART Output**
Signed Number	0	Signed Decimal Number
Any Value	1	"Error: Invalid character\n"
Any Value	2	"Error: Invalid expression\n"
Any Value	3	"Error: Input overflow\n"
Any Value	4	"Error: Output overflow\n"
Any Value	≥5	"Error: Unknown error code\n"

The only error your program needs to check for is i*nvalid characters* (`$a1` = 1), which are when a character is not a number character or one of the 3 operators (you can also allow space characters, but this is not required). In order to use the output function you PLP program needs to have initialized the stack pointer (`$sp`). You should typically initialize the stack pointer to `0x10fffffc` (the last address of RAM). The output is called using the following instruction (the line following this instruction must be a valid PLP instruction otherwise you will most likely get an error during simulation):

```
call project3_output_number
```

After this function call, the value in `$a0` (unless `$a1` is set to something other than 0) will be displayed on the UART and then the program will return to the instruction immediately following the *call* instruction.

Deliverables:

1. Submit a **PLP program** on Blackboard (16 points)
2. Take the **Post-Project 4 Assessment** on Blackboard (4 points)