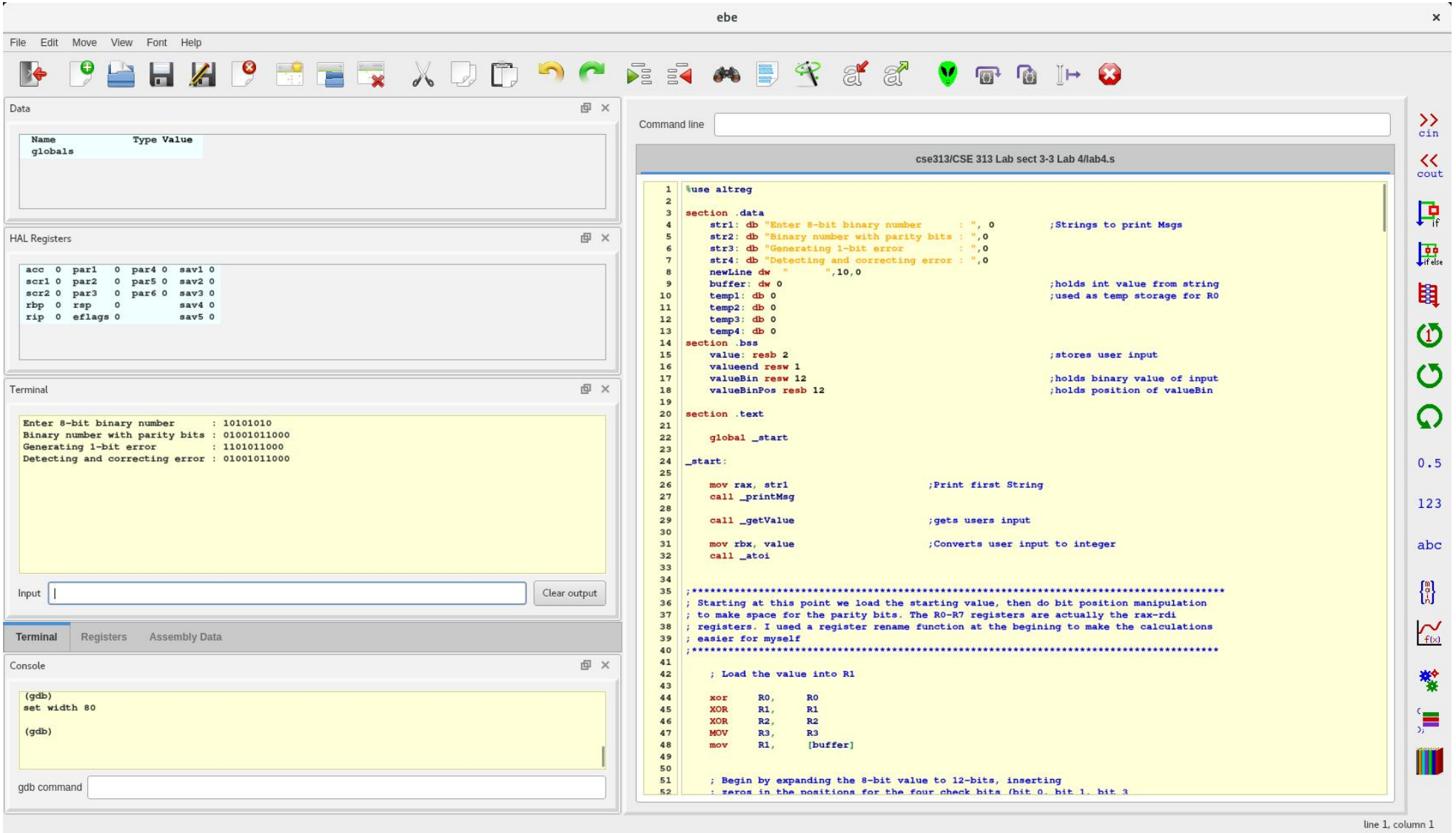
ebe
File Edit Move View Font Help
Data
Name Type Value
globals
HAL Registers
acc 0 par1 0 par4 0 sav1 0
scr1 0 par2 0 par5 0 sav2 0
scr2 0 par3 0 par6 0 sav3 0
rbp 0 rsp 0 sav4 0
rip 0 eflags 0 sav5 0
Terminal
Enter 8-bit binary number : 10101010
Binary number with parity bits : 01001011000
Generating 1-bit error : 1101011000
Detecting and correcting error : 01001011000
Input
Clear output
Terminal Registers Assembly Data
Console
(gdb)
set width 80
(gdb)
gdb command
Command line
cse313/CSE 313 Lab sect 3-3 Lab 4/lab4.s
%use altreg
section .data
str1: db "Enter 8-bit binary number : ", 0 ;Strings to print Msgs
str2: db "Binary number with parity bits : ",0
str3: db "Generating 1-bit error : ",0
str4: db "Detecting and correcting error : ",0
newLine dw " ",10,0
buffer: dw 0 ;holds int value from string
temp1: db 0 ;used as temp storage for R0
temp2: db 0
temp3: db 0
temp4: db 0
section .bss
value: resb 2 ;stores user input
valueend resw 1
valueBin resw 12 ;holds binary value of input
valueBinPos resb 12 ;holds position of valueBin
section .text
global _start
_start:
mov rax, str1 ;Print first String
call _printMsg
call _getValue ;gets users input
mov rbx, value ;Converts user input to integer
call _atoi
;*****************************************************************************************
; Starting at this point we load the starting value, then do bit position manipulation
; to make space for the parity bits. The R0-R7 registers are actually the rax-rdi
; registers. I used a register rename function at the begining to make the calculations
; easier for myself
;*****************************************************************************************
; Load the value into R1
xor R0, R0
XOR R1, R1
XOR R2, R2
MOV R3, R3
mov R1, [buffer]
; Begin by expanding the 8-bit value to 12-bits, inserting
; zeros in the positions for the four check bits (bit 0, bit 1, bit 3
cin
cout
if
if else
0.5
123
abc
f(x)
line 1, column 1