Week 6

* Assembly
  + Sign extension
    - 22
    - + EC
      * E = 1110
      * So the actual addition is
      * 0x00 00 00 22
      * + FF FF FF EC
      * = 00 00 00 0**E**
    - So you jump backwards! Woah!!
    - If the first bit of the hex val is 1 then we’re jumping backwards
    - forwards otherwise
  + Debugging with gdb

|  |  |
| --- | --- |
| **Command** | **Description** |
| help | List gdb command topics. |
| help *topic-classes* | List gdb command within class. |
| help *command* | Command description.  eg help show to list the show commands |
| apropos *search-word* | Search for commands and command topics containing *search-word*. |
| info args  i args | List program command line arguments |
| info breakpoints | List breakpoints |
| info break | List breakpoint numbers. |
| info break *breakpoint-number* | List info about specific breakpoint. |
| info watchpoints | List breakpoints |
| info registers | List registers in use |
| info threads | List threads in use |
| info set | List set-able option |
| **Break and Watch** |  |
| break *funtion-name*  break *line-number*  break *ClassName::functionName* | Suspend program at specified function of line number. |
| break +*offset*  break -*offset* | Set a breakpoint specified number of lines forward or back from the position at which execution stopped. |
| break *filename:function* | Don't specify path, just the file name and function name. |
| break *filename:line-number* | Don't specify path, just the file name and line number.  break *Directory/Path/filename*.cpp:62 |
| break \**address* | Suspend processing at an instruction address. Used when you do not have source. |
| break *line-number* if *condition* | Where condition is an expression. i.e. x > 5  Suspend when boolean expression is true. |
| break *line* thread *thread-number* | Break in thread at specified line number. Use info threads to display thread numbers. |
| tbreak | Temporary break. Break once only. Break is then removed. See "break" above for options. |
| watch *condition* | Suspend processing when condition is met. i.e. x > 5 |
| clear  clear *function*  clear *line-number* | Delete breakpoints as identified by command option.  Delete all breakpoints in *function*  Delete breakpoints at a given line |
| delete  d | Delete all breakpoints, watchpoints, or catchpoints. |
| delete *breakpoint-number*  delete *range* | Delete the breakpoints, watchpoints, or catchpoints of the breakpoint ranges specified as arguments. |
| disable *breakpoint-number-or-range*  enable *breakpoint-number-or-range* | Does not delete breakpoints. Just enables/disables them.  Example:  Show breakpoints: info break  Disable: disable 2-9 |
| enable *breakpoint-number* once | Enables once |
| continue  c | Continue executing until next break point/watchpoint. |
| continue *number* | Continue but ignore current breakpoint *number* times. Usefull for breakpoints within a loop. |
| finish | Continue to end of function. |
| **Line Execution** |  |
| step  s  step *number-of-steps-to-perform* | Step to next line of code. Will step into a function. |
| next  n  next *number* | Execute next line of code. Will not enter functions. |
| until  until *line-number* | Continue processing until you reach a specified line number. Also: function name, address, filename:function or filename:line-number. |
| info signals  info handle  handle *SIGNAL-NAME* *option* | Perform the following option when signal recieved: nostop, stop, print, noprint, pass/noignore or nopass/ignore |
| where | Shows current line number and which function you are in. |
| **Stack** |  |
| backtrace  bt  bt *inner-function-nesting-depth*  bt -*outer-function-nesting-depth* | Show trace of where you are currently. Which functions you are in. Prints stack backtrace. |
| backtrace full | Print values of local variables. |
| frame  frame *number*  f *number* | Show current stack frame (function where you are stopped)  Select frame number. (can also user up/down to navigate frames) |
| up  down  up *number*  down *number* | Move up a single frame (element in the call stack)  Move down a single frame  Move up/down the specified number of frames in the stack. |
| info frame | List address, language, address of arguments/local variables and which registers were saved in frame. |
| info args  info locals  info catch | Info arguments of selected frame, local variables and exception handlers. |
| **Source Code** |  |
| list  l  list *line-number*  list *function*  list -  list *start#,end#*  list *filename:function* | List source code. |
| set listsize *count*  show listsize | Number of lines listed when list command given. |
| directory *directory-name*  dir *directory-name*  show directories | Add specified directory to front of source code path. |
| directory | Clear sourcepath when nothing specified. |
| **Machine Language** |  |
| info line  info line *number* | Displays the start and end position in object code for the current line in source.  Display position in object code for a specified line in source. |
| disassemble *0xstart 0xend* | Displays machine code for positions in object code specified (can use start and end hex memory values given by the info line command. |
| stepi  si  nexti  ni | step/next assembly/processor instruction. |
| x *0xaddress*  x/nfu *0xaddress* | Examine the contents of memory.  Examine the contents of memory and specify formatting.   * n: number of display items to print * f: specify the format for the output * u: specify the size of the data unit (eg. byte, word, ...)   Example: x/4dw var |
| **Examine Variables** |  |
| print *variable-name*  p *variable-name*  p *file-name::variable-name*  p '*file-name*'::*variable-name* | Print value stored in variable. |
| p \**array-variable*@*length* | Print first # values of array specified by *length*. Good for pointers to dynamicaly allocated memory. |
| p/x *variable* | Print as integer variable in hex. |
| p/d *variable* | Print variable as a signed integer. |
| p/u *variable* | Print variable as a un-signed integer. |
| p/o *variable* | Print variable as a octal. |
| p/t *variable*  x/b *address*  x/b &*variable* | Print as integer value in binary. (1 byte/8bits) |
| p/c *variable* | Print integer as character. |
| p/f *variable* | Print variable as floating point number. |
| p/a *variable* | Print as a hex address. |
| x/w *address*  x/4b &*variable* | Print binary representation of 4 bytes (1 32 bit word) of memory pointed to by address. |
| ptype *variable*  ptype *data-type* | Prints type definition of the variable or declared variable type. Helpful for viewing class or struct definitions while debugging. |
| **GDB Modes** |  |
| set *gdb-option* *value* | Set a GDB option |
| set logging on  set logging off  show logging  set logging file *log-file* | Turn on/off logging. Default name of file is gdb.txt |
| set print array on  set print array off  show print array | Default is off. Convient readable format for arrays turned on/off. |
| set print array-indexes on  set print array-indexes off  show print array-indexes | Default off. Print index of array elements. |
| set print pretty on  set print pretty off  show print pretty | Format printing of C structures. |
| set print union on  set print union off  show print union | Default is on. Print C unions. |
| set print demangle on  set print demangle off  show print demangle | Default on. Controls printing of C++ names. |
| **Start and Stop** |  |
| run  r  run *command-line-arguments*  run < *infile* > *outfile* | Start program execution from the beginning of the program. The command break main will get you started. Also allows basic I/O redirection. |
| continue  c | Continue execution to next break point. |
| kill | Stop program execution. |
| quit  q | Exit GDB debugger. |