How to make a new project for sid (arm-7) and ecos with Eclipse in Linux

Sid

You can download the source code of sid from http://sourceware.org/sid/ and install it.

- Download and extract the source.
- Compile the source:
  - $cd sid/src
  - $./configure –prefix=/sid_installation_path
  - $make
- Install sid:
  - $sudo make install

Some issues about sid building:

- If tk component is not installed correctly you may have to remake and reinstall tk component.
  - $cd src/sid/component/tk
  - $make
  - $sudo make install
- You may also have to ./configure first in src/tk directory BEFORE executing ./configure in /src directory.

However, a modified version of sid is available at https://github.com/adritheone/SID-eCos. This version has a hexadecimal keyboard.

Executing SID

To execute sid, a configuration file must be edited. ($sid file.conf). To create the basic configuration file for a ARM PID7 little endian to execute the binary file “example”, the following instruction must be executed:

$amr-elf-sid --cpu arm --board=pid7t-normalmap --gdb=2000 -EL --tksm example –no-run

Configuration file saved to `example.conf'.

Execution:

$sid example.conf

Download and install eCos libraries.

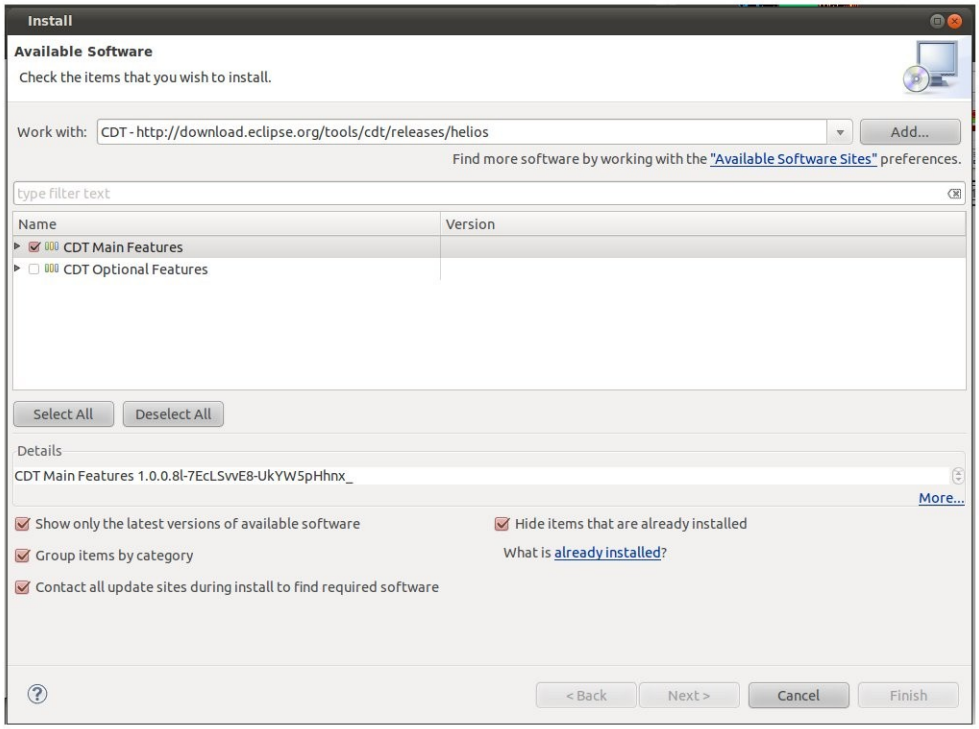
Ecos and the full instructions for installation can be found at http://ecos.sourceware.org/:

- Download eCos installer: $wget --passive-ftp ftp://ecos.sourceware.org/pub/ecos/ecos-install.tcl
- Installation: $sh ecos-install.tcl
  - Note that ARM-EABI toolchain has to be installed
- Download configuration file from repository: wget... arm_pid.ecc
- Build the library
  - cd ecos-3.0/tools/bin
  - ./configtool ../(...)/arm_pid.ecc
  - make sure the build path (Tools->Paths->Build Path) is set to the toolchain path (…/ecos/gnutools/arm-eabi/bin)
  - Build the library: Build->Library

This will create the necessary eCos libraries under the name of arm_pid_build and arm_pid_install (in the same directory where arm_pid.ecc is saved). However, a built library is available at https://github.com/adritheone/SID-eCos.

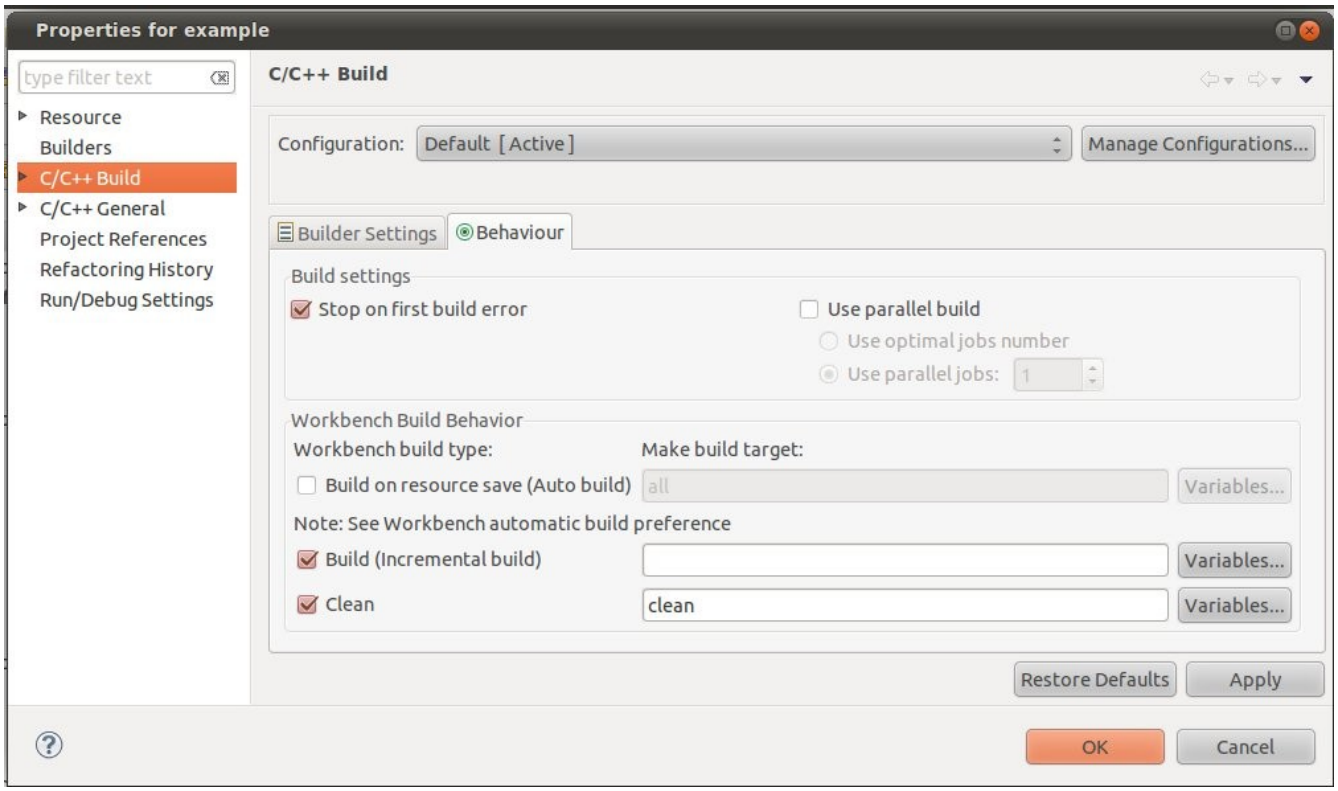
Download Eclipse IDE

- Download Eclipse Helios 3.6 from http://www.eclipse.org/helios/
- Install CDT:
  - Help->Install new software
  - Add repository: http://download.eclipse.org/tools/cdt/releases/helios
  - Install CDT

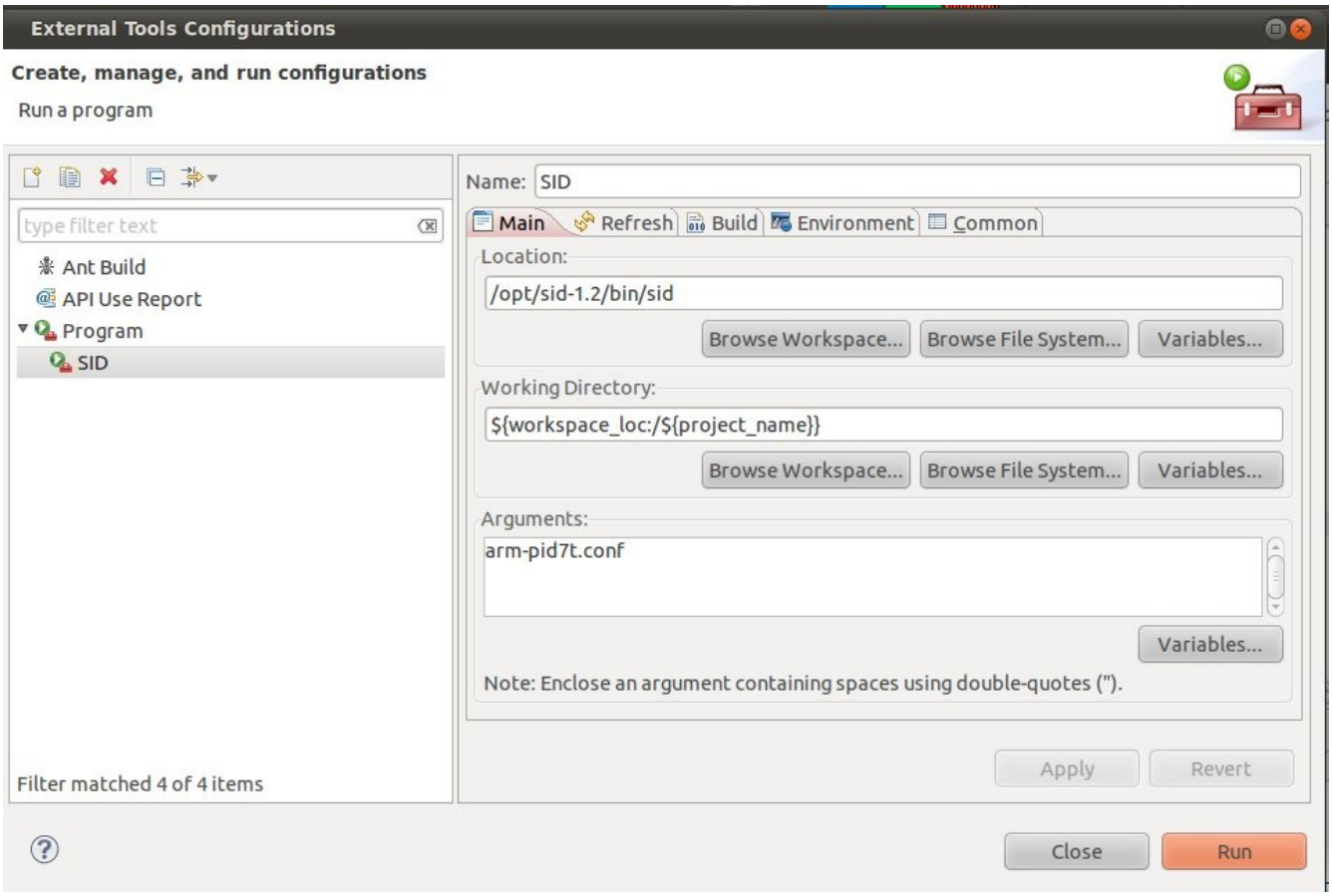

Install
Available Software
Check the items that you wish to install.
Work with:
CDT - http://download.eclipse.org/tools/cdt/releases/helios
Add...
Find more software by working with the "Available Software Sites" preferences.
type filter text
Name
Version
CDT Main Features
CDT Optional Features
Select All
Deselect All
Details
CDT Main Features 1.0.0.8l-7EcLSvvE8-UkYW5pHhnx_
More...
Show only the latest versions of available software
Hide items that are already installed
Group items by category
What is already installed?
Contact all update sites during install to find required software
< Back
Next >
Cancel
Finish

Configure eclipse

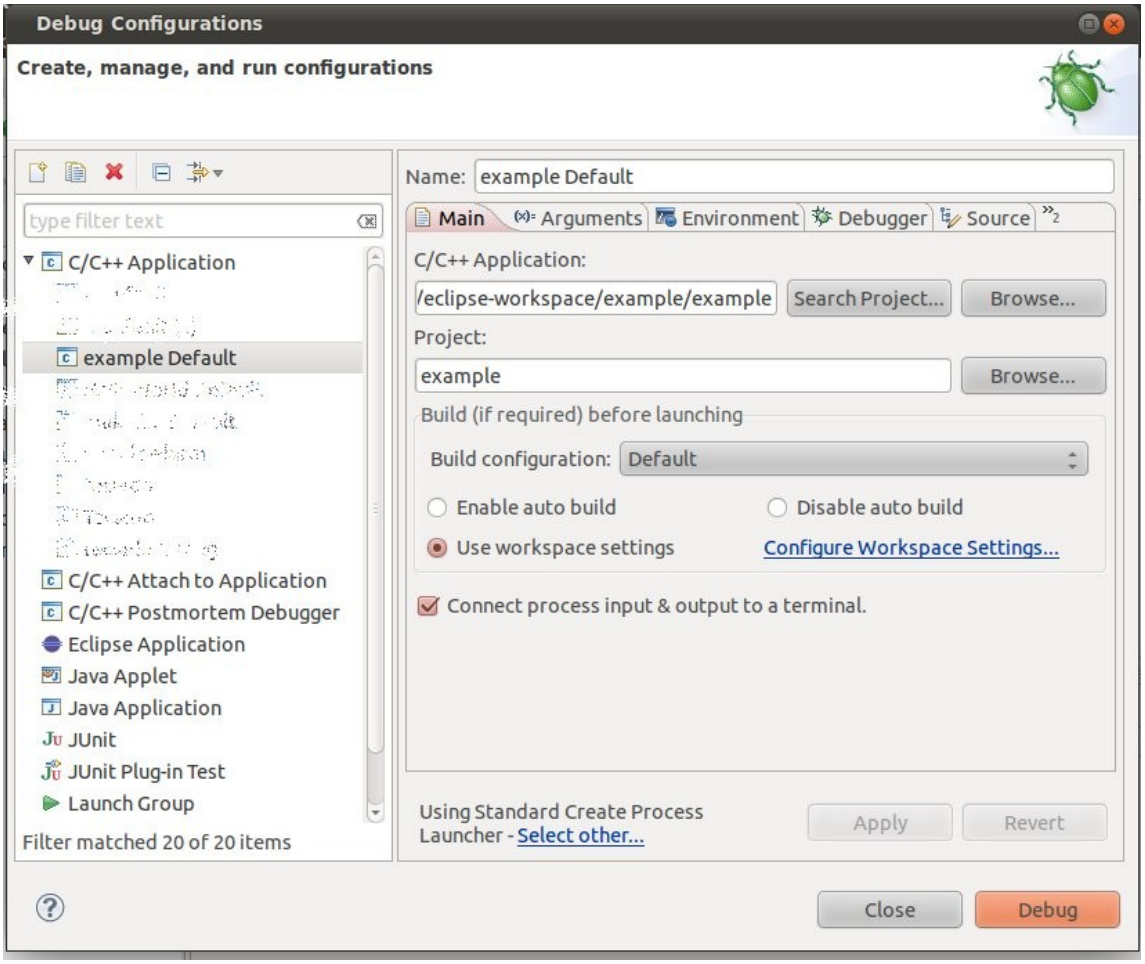
- Create a new C Makefile Project example
  - Import the Makefile and Make.params from (https://github.com/adritheone/SID-eCos)
  - Change the Make.params line to the eCos libraries path:
    - `export PREFIX := …/arm_pid_install`
  - Change the Makefile lines and include your program's name:
    - ```
      SRCS=example.c
      OBJS=${SRCS:%.c=%.o}
      DST=example
      ```
- Create the sid configuration file
  - $amr-elf-sid --cpu arm --board=pid7t-normalmap --gdb=2000 -EL --tksm example –no-run
  - Or import arm-pid7t.conf (from https://github.com/adritheone/SID-eCos) and change line 187 to include your program's name
    - set cpu-loader file "example"
- Configure building
  - Project->Properties->C/C++ Build (Behaviour tap)
    - Build(incremental build): delete 'all'


      ```

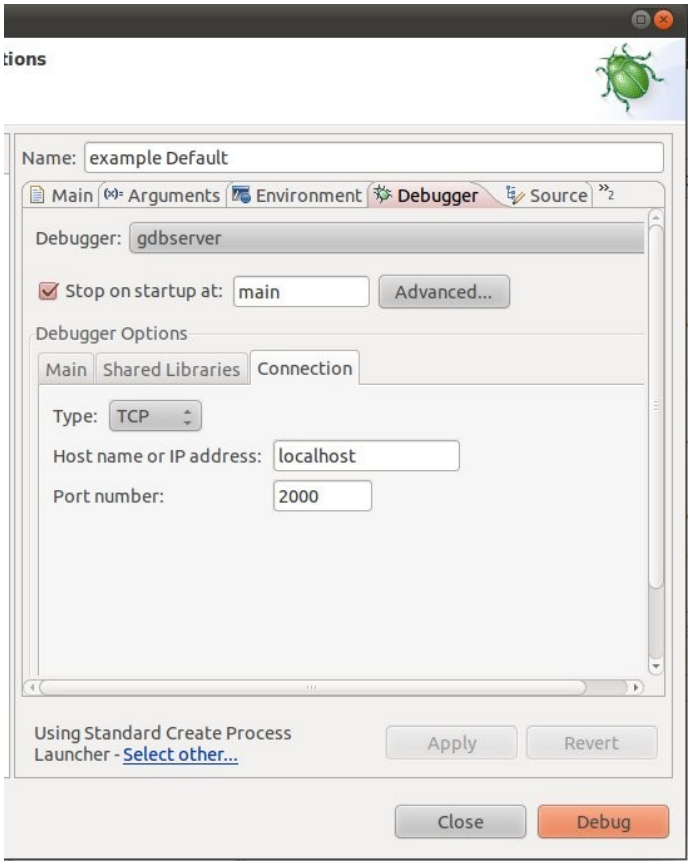
- Configure execution
  - Run->Run external tool->External tool configuration
  - Create a new Program called SID
    - Location: /path_to_sid_installation/bin/sid
    - Working Directory: ${workspace_loc:/${project_name}}
    - Arguments:arm-pid7t.conf (or the name of your sid configuration file)



- Configure debug: We will launch a gdbserver on port 2000. If you have created your own sid configuration file, make sure the port is set properly.
  - Run->Debug Configurations
  - Create new C/C++ Aplication
    - Main
      - C/C++ Aplication: …/eclipse-workspace/example/example (or the path to your program)
      - Project: name of the project



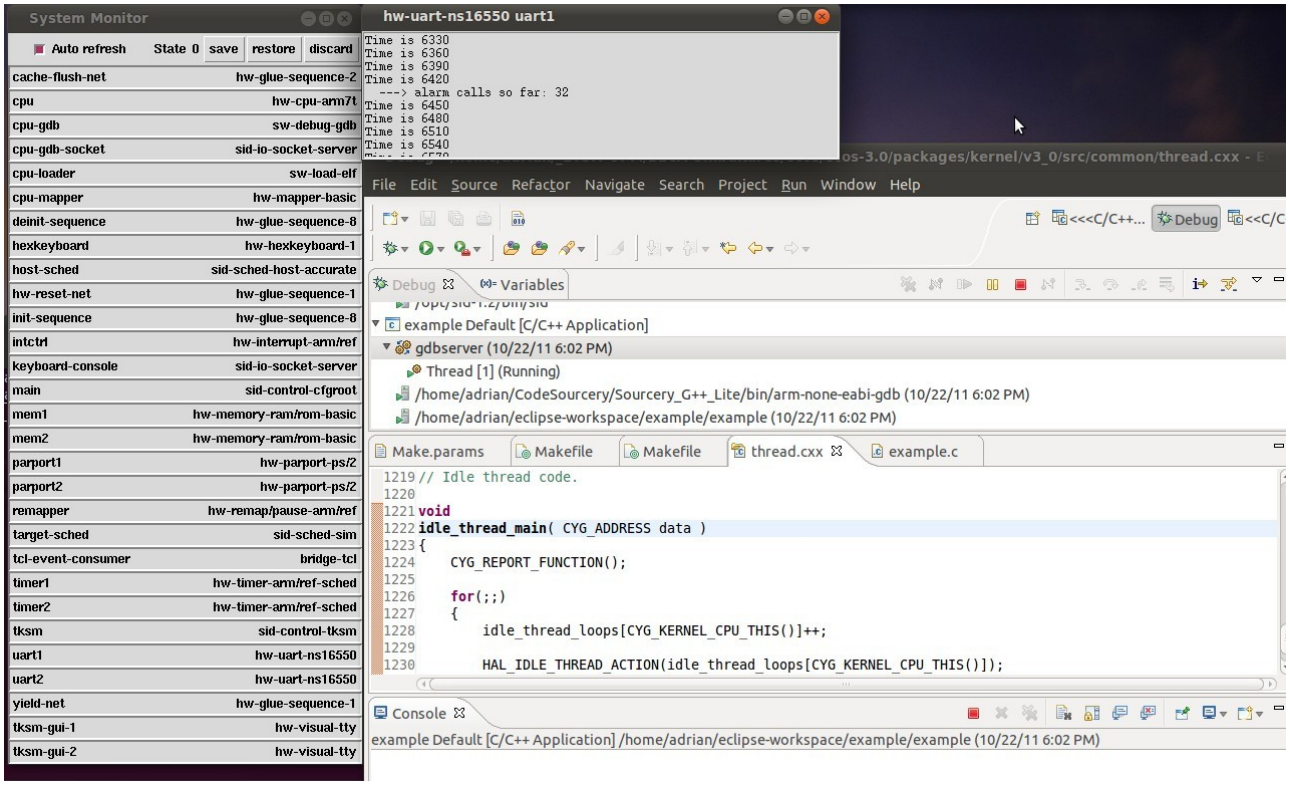
- Debugger
  - Using Standard Create Process Launcher
  - Debugger: gdbserver
  - Stop on startup at: main (or cyg_user_start)
  - Debugger Options: Main
    - GDB debugger: /...path to arm-eabi toolchain.../bin/arm-eabi-gdb
  - Debugger Options: Connection
    - Type: TCP
    - Host name: localhost (or 127.0.0.1)
    - Port: 2000 (or whatever port you have chosen in sid configuration file)
- Source
  - The source code of your program will be already added, but it is also very useful to add the eCos libraries (…/arm_pid_install/include).



Some more issues

- Codesourcery toolchain can be used in stead of the one provided by eCos. Just make sure that it is accessible at PATH variable.
    - To use Codesourcery toolchain to compile, change Make.params variable:

      export COMMAND_PREFIX := arm-eabi- to export COMMAND_PREFIX := arm-none-eabi
    - To use Codesourcery toolchain to debug, just put the correct path into GDB Debugger at Debugger configuration.
- Note that only one instance of sid can be open at the same time. Otherwise, the TCP port used for debugging will be unaccessible. So, once you have finished debugging, close both the debug server and the sid program form eclipse (or do $kill all sid).
- The standard output is set to the uart1 of sid. Therefore, when sid starts open the uart1 gui to see the program output.


System Monitor
Auto refresh
State 0
save
restore
discard
cache-flush-net hw-glue-sequence-2
cpu hw-cpu-arm7t
cpu-gdb sw-debug-gdb
cpu-gdb-socket sid-io-socket-server
cpu-loader sw-load-elf
cpu-mapper hw-mapper-basic
deinit-sequence hw-glue-sequence-8
hexkeyboard hw-hexkeyboard-1
host-sched sid-sched-host-accurate
hw-reset-net hw-glue-sequence-1
init-sequence hw-glue-sequence-8
intctrl hw-interrupt-arm/ref
keyboard-console sid-io-socket-server
main sid-control-cfgroot
mem1 hw-memory-ram/rom-basic
mem2 hw-memory-ram/rom-basic
parport1 hw-parport-ps/2
parport2 hw-parport-ps/2
remapper hw-remap/pause-arm/ref
target-sched sid-sched-sim
tcl-event-consumer bridge-tcl
timer1 hw-timer-arm/ref-sched
timer2 hw-timer-arm/ref-sched
tksm sid-control-tksm
uart1 hw-uart-ns16550
uart2 hw-uart-ns16550
yield-net hw-glue-sequence-1
tksm-gui-1 hw-visual-tty
tksm-gui-2 hw-visual-tty
hw-uart-ns16550 uart1
Time is 6330
Time is 6360
Time is 6390
Time is 6420
---> alarm calls so far: 32
Time is 6450
Time is 6480
Time is 6510
Time is 6540
os-3.0/packages/kernel/v3_0/src/common/thread.cxx - E
File Edit Source Refactor Navigate Search Project Run Window Help
<<<C/C++...
Debug
<<C/C
Debug
Variables
example Default [C/C++ Application]
gdbserver (10/22/11 6:02 PM)
Thread [1] (Running)
/home/adrian/CodeSourcery/Sourcery_G++_Lite/bin/arm-none-eabi-gdb (10/22/11 6:02 PM)
/home/adrian/eclipse-workspace/example/example (10/22/11 6:02 PM)
Make.params
Makefile
Makefile
thread.cxx
example.c
1219 // Idle thread code.
1220
1221 void
1222 idle_thread_main(CYG_ADDRESS data)
1223 {
1224 CYG_REPORT_FUNCTION();
1225
1226 for(;;)
1227 {
1228 idle_thread_loops[CYG_KERNEL_CPU_THIS()]++;
1229
1230 HAL_IDLE_THREAD_ACTION(idle_thread_loops[CYG_KERNEL_CPU_THIS()]);
Console
example Default [C/C++ Application] /home/adrian/eclipse-workspace/example/example (10/22/11 6:02 PM)