INE5412-04208A (20151) - Sistemas Operacionais I

Information about Developing embedded software with EPOS and EposMotes II

Embedded Parallel Operating System (EPOS)

The EPOS Project (Embedded Parallel Operating System) aims at automating the development of embedded systems so that developers can concentrate on what really matters: their applications. EPOS relies on the Application-Driven Embedded System Design(external link) (ADESD) method to guide the development of both software and hardware components that can be automatically adapted to fulfill the requirements of particular applications. EPOS features a set of tools to support developers in selecting, configuring, and plugging components into its application-specific framework. The combination of methodology, components, frameworks, and tools enable the automatic generation of an application-specific embedded system instances.

How to Use Embedded Parallel Operating System (EPOS for the Final Project of Operating Systems I

1. Know the EPOS licence.
   EPOS Software License v1.0.
   A copy of this license is available at the EPOS system source tree root.
   A copy of this license is also available online a:
   http://epos.lisha.ufsc.br/EPOS+Software+License+v1.0
   Note that EPOS Software License applies to both source code and executables.

2. Read more about EPOS on

http://epos.lisha.ufsc.br/HomePage
http://epos.lisha.ufsc.br/EPOS+User+Guide

You also should download extra matterial made available on
http://www.inf.ufsc.br/~cancian/FPinfos.tar.gz

vigatio

inistrati

3. From that same webpage (epos.lisha.ufsc.br), access link bellow and follow instructions there to download and install EPOS
http://epos.lisha.ufsc.br/EPOS+Software

You'll need to register yourself so you can download OpenEPOS-1.1 and GCC4.4.4 for ARM architectures.

You must agree with licence on
http://epos.lisha.ufsc.br/EPOS+Software+License+v1.0

4. You can install OpenEPOS on your linux OS or you can create a virtual machine to host it. UFSC provides a VM for students enrolled to INE5412. If you choose to use UFSC/Setic VM, proceed as follow:
Access idufsc
https://idufsc.ufsc.br/
Goto "Cloud" service and then "Virtual Servers".
Create your own virtual server (VM) ad access ir using SSH.
Now you can proceed to EPOS instalation using that VM.
Compiling EPOS apps in the VM demandslater you'll need to copy the compiled application back to your physical machine so you can save it to the EposMoteII physically connected to your machine. A simple "scp" can solve that.

5. Update and install extra necesssary packages, by typing
sudo apt-get update
sudo apt-get -y upgrade
sudo apt-get -y install g++ gcc build-essential gdb gdb-multiarch subversion qemu qemu-system-arm qemu-utils tcsh

6. Uncompress OpenEPOS-1.1 RC.tgz on a proper path of your home dir, and uncompress arm-gcc-4.4.4.tar.gz on /usr/local/arm dir.
ARM gcc will be installed on /usr/local/arm/gcc-4.4.4. Create a symbolic link called "gcc" pointing to "gcc-4.4.4"

7. Follow steps available on http://epos.lisha.ufsc.br/EPOS+User+Guide to install and configure EPOS
Read the section "Common Installation Problems" (3.2.1)
At least you will need /bin/sh links to bash instead of dash.
If you have choosen to use UFSC/Setic VM, remember it is a 64 bits Linux, so you'll need install some extra packages, as described in the webpage (for Ubuntu 14.04)
To compile apps for EposMotes II, makedefs should in mode library, for arch arm7 and machine mc13224v.

Undo

8. That's it. Try to build the default app for EposMotesII just typing
make all
or
make APPLICATION=mc13224v_app
(You may want to clean up evething before compiling. Do it by typing "make veryclean")

You you have followed all previous steps, compilation/building should complete successfuly and therefore the following binay file should be created:
img/mc13224v_app.img

Now you just have to debug and/or to program the physical EposMoteII plataform with this binary image

9. To debug your application you can use "qemu" and optionally you can enable gdb integration by setting "GDB_DEBUG := 1" on makedefs file.

See section "3.5. Running" and "3.5.1.1. Virtual Machine QEMU" for testing your application.

10. You can know more about the physical platform of EposMotes II on
http://epos.lisha.ufsc.br/EPOSMote+II and related webpages.

11. To program EposMotesII physical platform, just read the section "3.5.5. ARM7" and its subsection "Programming EPOSMoteII through the serial port" on
http://epos.lisha.ufsc.br/EPOS+User+Guide

Since python script "red-bsl" is store in the "bin/" dir and it needs other script from the same dir, it's better you go into bin dir (cd bin) and then type
sudo python red-bsl.py -t /dev/ttyUSB0 -f ../img/mc13224v_app.bin -S
The -S parameter saves the app into the non-volatile flash memory instead of the RAM memory.

12. Now you should be able to develop new applications using OpenEPOS for the EposMoteII and to program that platform with your new application. You may know how to configure Epos by reading section "3.6. Configurability" on
http://epos.lisha.ufsc.br/EPOS+User+Guide

and finally you really should read section "4. EPOS Programming" to know EPOS Application Programming Interface (API) and therefore to be able to develop new application using EPOS abstractions.

If you need to go down into the OS layer, ie, it's not enough to create an application, but you need to improve an existing or create a new OS component, then you should read de "EPOS Developer" support matterial.

Information about the problem writing the FLASH (return from EPOS Support team):

Se há algo na flash, não há como carregar outra imagem, nem mesmo na RAM. Há três maneiras de apagar a flash:

1) Se o firmware gravado for EPOS, o EPOS verifica o pino do botão SW1 da outra placa está pressionado durante o boot e "se auto-apaga" (apertar reset segurando o botão branco da placa). Mas, isto requer que o EPOS chegue no Machine::init(). Se alguém tiver mexido algo na inicialização do EPOS, ele pode não estar chegando a esse ponto. Neste caso, o apagamento tem que ser físico:

2) Apagamento através do jumpers J4 e J5. Tire o mote da USB, coloque os dois jumpers na posição 2-3, ligue o mote por uns 2 s, desligue o mote, coloque os jumpers de volta na posição 1-2, e tudo deve funcionar normalmente. É muito raro, mas pode ser que isto falhe:

3) Se 1 e 2 falharem, será necessário recuperar a memória carregando um código "desgravador" via jtag. Nesse caso, volte a conversar com o professor.

Há algumas informações sobre isto na página do EPOS: http://epos.lisha.ufsc.br/EPOS+User+Guide#EPOSMoteII_MC13224V_

Last modified: segunda, 17 agosto 2015, 10:42
