Universidade Federal do Rio Grande do Sul
Escola de Engenharia
Departamento de Sistemas Elétricos de Automação e Energia
ENG10032 Microcontroladores

Roteiro de Laboratório 13
Yocto

Prof. Walter Fetter Lages

26 de novembro de 2019

1 Objetivo

O objetivo deste laboratório é criar uma imagem para ser carregada no cartão microSD, a partir do Yocto. Todos os arquivos binários da distribuição serão compilados a partir do seu código fonte, de forma que no futuro se possa introduzir modificações no *software* instalado na Galileo, incluindo o próprio *kernel*.

2 Fundamentação Teórica

As imagens para o microSD são criadas a partir do Yocto. O Yocto é um projeto para desenvolver ferramentas para criar distribuições Linux personalizáveis para sistemas embarcados independentemente da arquitetura de *hardware*. O Poky é o sistema de referência (um conjunto de ferramentas e metadados que formam um exemplo funcional) do projeto Yocto e pode ser baixado e usado para criar uma distribuição personalizada.

O Poky utiliza um *framework* desenvolvido pelo projeto OpenEmbedded para criar a distribuição. Este *framework* depende de *scripts* denominados *Board Support Package* (BSP), que especificam os detalhes de cada placa suportada pelo Yocto. O Bitbake é um utilitário que interpreta os arquivos BSP e gera uma imagem Poky personalizada. Isso implica baixar e compilar todo o *software* da distribuição. Embora uma imagem típica tenha um pouco menos de 2GB, o processo todo requer em torno de 60GB de espaço em disco, devido aos arquivos temporários gerados. Para compilação da imagem é necessário um sistema Linux de 64 bit.

3 Experimentos

1. Baixe os arquivos do Poky para a Galileo do Moodle[1]. O nome do arquivo é `iot-devkit-yp-poky-galileo-20160606.zip`. Salve o arquivo antes de tentar descompactá-lo.

2. Descompacte o arquivo e entre no diretório criado.

```
unzip iot-devkit-yp-poky-galileo-20160606.zip
ln -s iot-devkit-yp-poky-galileo-20160606 iot-devkit
cd iot-devkit
```

3. Limpe a variável de ambiente `PATH` com o comando[2]:

```
PATH=/bin:/usr/bin:/usr/sbin
```

4. O diretório temporário usado pelo Poky não pode estar em uma partição montada por NFS, como nas máquinas do laboratório. Portanto, edite o arquivo `build_galileo/conf/auto.conf` e altere ou inclua a seguinte linha[3]:

```
TMPDIR="/usr/local/tmp/poky"
```

5. Configure as variáveis de ambiente necessárias para o Poky executando o *script*:

```
cd poky
source oe-init-build-env ../build_galileo/
```

   Ao final da execução do *script* o diretório corrente será `build_galileo/`.

6. Crie a distribuição com o comando:

```
bitbake -k iot-devkit-prof-dev-image
```

   Enquanto a distribuição é gerada, aproveite o tempo para concluir tarefas pendentes de laboratórios anteriores.

7. Quando o `bitbake` tiver terminado a compilação, crie uma imagem da distribuição com os comandos:

[1] Ou do site da Intel <http://iotdk.intel.com/src/3.5/galileo/iot-devkit-yp-poky-galileo-20160606.zip>.

[2] Esta configuração é válida apenas para o terminal atual. Se ele for fechado e outro aberto, será necessário ajustar a variável de ambiente novamente.

[3] Isso só é necessário porque os diretórios de usuário nas máquinas de laboratório são montados por NFS.

```
cd ../poky/meta-intel-iot-devkit/scripts
./wic_monkey create -e iot-devkit-prof-dev-image lib/image/canned-wks/iot-devkit.wks -o toMicroSD
cd ../../../build_galileo
```

8. Instale a imagem no cartão microSD com o comando:

```
dd if=toMicroSD/build/iot-devkit-*.direct of=/dev/sdb BS=3M conv=fsync
```

9. Instale o cartão microSD na Galileo e verifique se ele está funcionando.