



embedded. Computing.



• Toradex
Swiss. Embedded. Computing.



toradex

Embedded. Computing.



Introdução ao desenvolvimento de sistemas embarcados com abordagem multicore assimétrica

Toradex

- Fundada em 2003 na Suíça
- Operação Global com escritório de engenharia e vendas:
Brasil, USA, China, Índia, Japão e Vietnã
- Computadores em Módulo com processadores ARM
- Famílias de Módulos
 - Colibri (67.6x36.7mm)
 - Vybrid (VF50 e VF61)
 - iMX6
 - Apalis (82x45mm)
 - iMX6

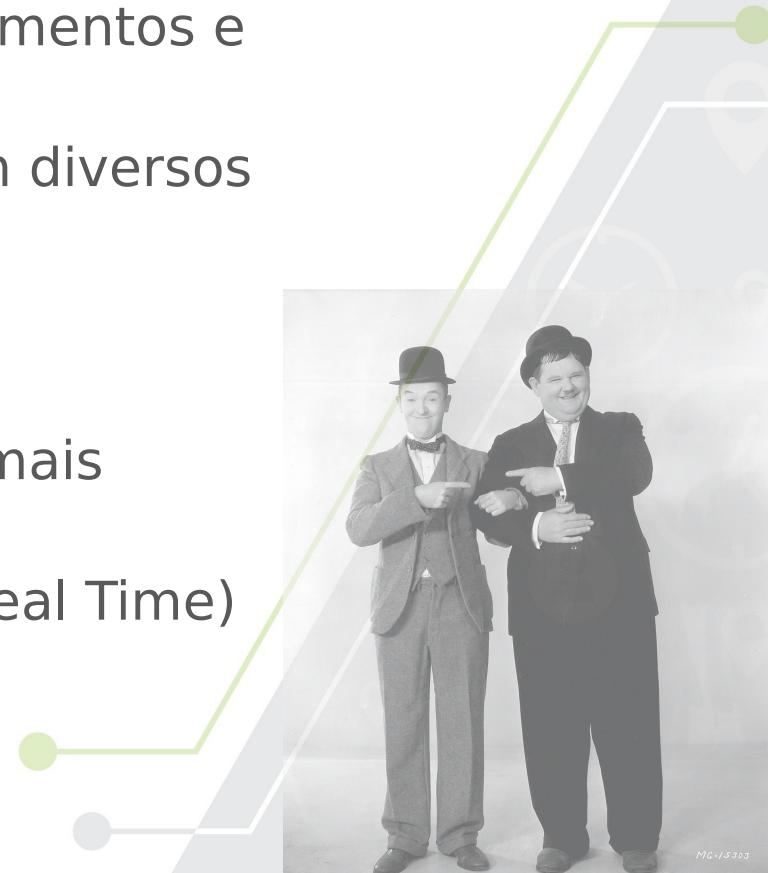


Raul Rosetto Muñoz

- Técnico Eletrônica – SENAI
 - Microcontrolador PIC.
- Engenheiro Eletricista – Universidade São Francisco
 - Venturus Centro de Inovação Tecnológica (2010 – 2011)
 - Microcontrolador PIC e MSP430
 - Phi Innovations (2011 – 2014)
 - Desenvolvimento de Projetos de sistemas embarcados utilizando LINUX
- Toradex (2014 – Atual) – 3 meses na Toradex Suíça
 - Engenharia de Aplicação e Vendas
 - Desenvolvimento de Linux Toradex - Global

Multicore Assimétrico

- Diferentes processadores no mesmo chip
 - Compartilham os mesmos barramentos e dispositivos.
- **Multicore Simétrico** é comum em diversos dispositivos
 - PC, celulares, tablets
 - Dispositivos embarcados
- **Multicore Assimétrico** pode ser mais eficiente para diversos cenários
 - Sistemas de tempo real (Hard Real Time)
 - CNC/3D plotter/2D plotter



Abordagem “Tradicional”

- Dois processadores separados
 - Processador rodando Linux (ou outro OS) voltado para aplicação.
 - Microcontrolador rodando RTOS.
- Comunicação serial + Sinais adicionais para gerar interrupções.
- Não compartilham memória.
 - Quantidade limitada de memória para a MCU externa.
- Não compartilham os dispositivos (UART, I2C, CAN).

ARM

- Cortex-A
 - Media/Alta performance
 - Cache, FPU, NEON
 - Controle gráfico
 - Conectividade
 - Multimedia
 - Adequado para um sistema operacional completo
- Cortex-M MCU
 - Baixo custo
 - Adequado para firmware e RTOS

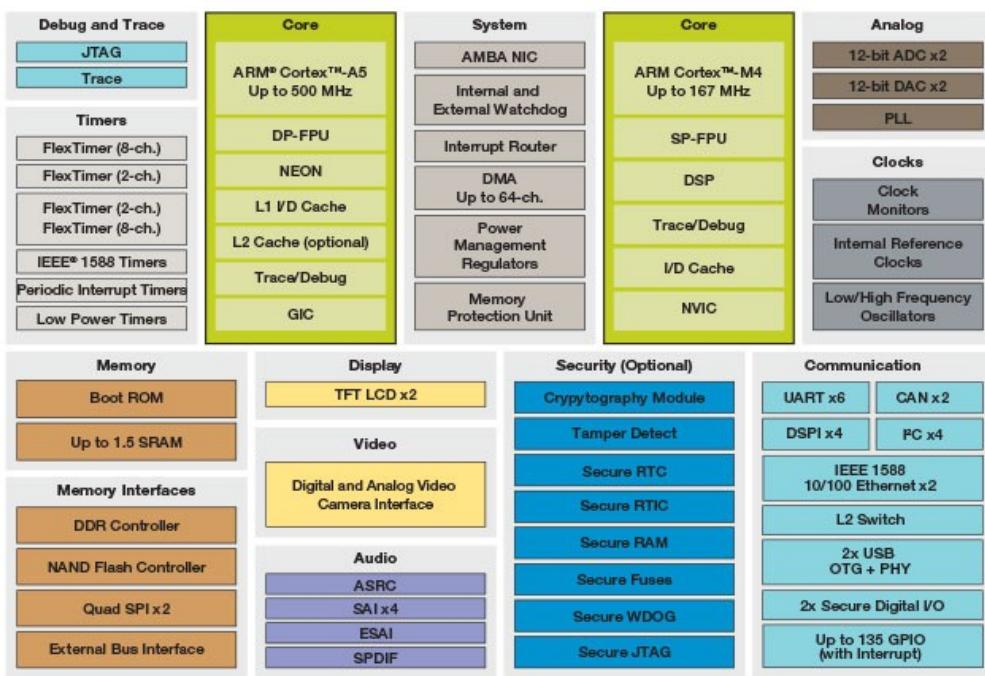


Vybrid Freescale

Família de processadores ARM voltados para dispositivos embarcados.

- VF3XX - Single core - sem DDR
- VF5XX - Single core - DDR
- VF6XX - Multi core - DDR

Vybrid VF6xx Block Diagram



Toradex Colibri



Colibri VF50
Cortex-A5 400MHz
128MB FLASH
128MB RAM



Colibri VF61
Cortex-(A5-500/M4-167)MHz
512MB FLASH
256MB RAM



Colibri iMX6S
1x(Cortex-A9) 1GHz
4GB FLASH
256MB RAM



Colibri iMX6DL
2x(Cortex-A9) 1GHz
4GB FLASH
512MB RAM



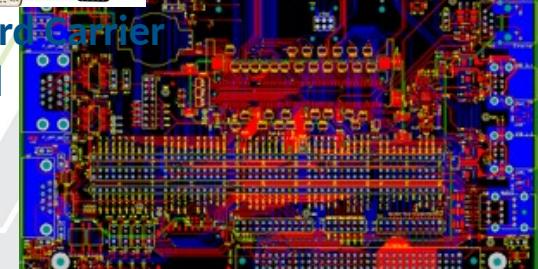
Iris Volume Carrier Board



Viola Volume Carrier Board



Evaluation Board



Exemplos Vybrid VF50 (A5)

- RDP (Remote Desktop Protocol)

Com um processador VF50/128RAM/128FLASH conecta-se por meio do protocolo RDP em um outro computador Windows.
http://www.youtube.com/v/E5Qj0_rWPNc



Exemplos Vybrid VF61 (A5+M4)

- XY Plotter

A5 (Linux)– Multimídia/Interface com Usuário/ Conectividade

M4 (ECOS) – Controle de Motores em Tempo Real

<http://www.youtube.com/embed/empqTT31boo>





ToolChain: Conjunto de ferramentas que interpretam e criam links do código para gerar um Binário executável para determinada arquitetura.

YOCTO/OpenEmbedded: Projeto Open Source com templates, ferramentas e métodos para ajudar na criação de um Linux personalizado para Sistemas Embarcados.

Toolchain

- 1) Compilar a própria Toolchain utilizando o OpenEmbedded.
- 2) Instalar a Toolchain por meio do arquivo já compilado:

```
angstrom-eglibc-x86_64-armv7at2hf-vfp-neon-toolchain-qte-v2014.06.sh  
Enter target directory for SDK (default: /usr/local/oecore-x86_64): Y
```

- 3) Aceite o caminho padrão.

A toolchain que vamos usar pode ser encontrada em:

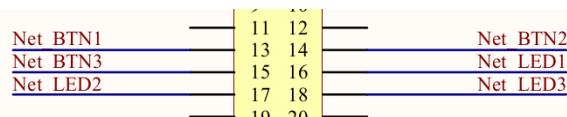
```
$ find /usr/local/oecore-x86_64/ -name "arm-angstrom-linux-gnueabi-*"  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-gcc-nm  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-nm  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-gcc-ranlib  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-g++  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-ar  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-gcc  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-ranlib  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-dwp  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-addr2line  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-as  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-readelf  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-size  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-objdump  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-strings  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-gcc-ar  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-c++filt  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-cpp  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-gprof  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-elfedit  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-gdb  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-gcov  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-strip  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-ld  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-objccopy  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-ld.gold  
/usr/local/oecore-x86_64/sysroots/x86_64-angstromsdk-linux/usr/bin/arm-angstrom-linux-gnueabi/arm-angstrom-linux-gnueabi-ld.bfd
```

Iris Carrier Board

IRIS EXTENSION HEADER



CAPE (3 botões + 3 LEDs)



Vybrid VF61

A5 (LINUX)

M4 (Firmware)

X1 Pin	GPIO Pad	GPIO Port	ALT0 (GPIO)	ALT1	ALT2	ALT3	ALT4	ALT5	ALT6	ALT7
98	PAD_46	PORT1[14]	PTC1	RMIIO_MDIO/ MII0_MDC	FTM1_CH1	SPI0_PCS2	ESAI_FST	SDHC0_CMD	VIU_DATA1	RCON19
133	PAD_88	PORT2[24]	PTD9	QSPI0_B_DATA3	SPI3_PCS1		FB_AD6		SAI1_TX_SYNC	DCU1_B0
103	PAD_48	PORT1[16]	PTC3	RMIIO_RXD1/ MII0_RXD[1]	SCI1_RX		ESAI_SDO1	SDHC0_DAT1	VIU_DATA3	DCU0_R0
101	PAD_47	PORT1[15]	PTC2	RMIIO_CRS_DV	SCI1_TX		ESAI_SDO0	SDHC0_DAT0	VIU_DATA2	RCON20
97	PAD_50	PORT1[18]	PTC5	RMIIO_RXER/ MII0_RXER	SCI1_CTS	SPI1_PCS0	ESAI_SDO3/ ESAI_SD12	SDHC0_DAT3	VIU_DATA5	DCU0_G0
85	PAD_53	PORT1[21]	PTC8	RMIIO_TXEN/ MII0_TXEN		SPI1_SCK			VIU_DATA8	DCU0_B1

First Steps - Iris Carrier Board

```
$ picocom -b 115200 /dev/tty<X>
```

```
raul@localhost:~
```

File Edit View Search Terminal Help

```
| [ ] | - | [ ] | [ ] | [ ] |
```

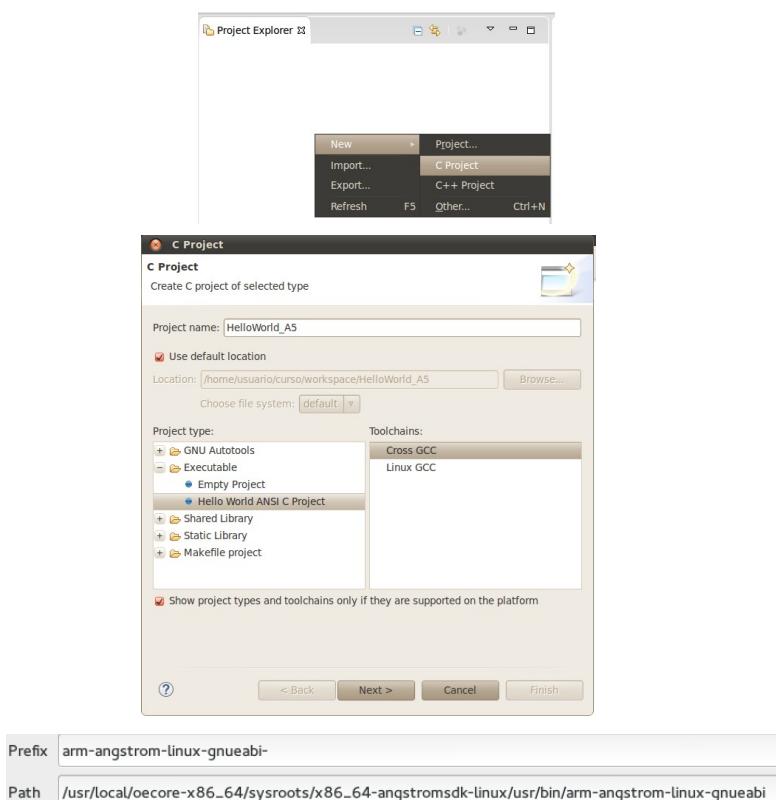
```
The Angstrom Distribution colibri-vf ttyLP0
Angstrom v2014.06 - Kernel 3.18.5-v2.3b7+gc78b5ae
Colibri_VF_LinuxImage-qte-xyplotterV2.2_20150304

colibri-vf login: root
Last login: Wed Mar 4 15:25:42 UTC 2015 from 192.168.10.1 on pts/1
root@colibri-vf:~# echo 47 > /sys/class/gpio/export
root@colibri-vf:~# echo out > /sys/class/gpio/gpio47/direction
root@colibri-vf:~# echo 1 > /sys/class/gpio/gpio47/value
root@colibri-vf:~# echo 0 > /sys/class/gpio/gpio47/value
root@colibri-vf:~# df -h
Filesystem      Size   Used Available Use% Mounted on
ubi0:rootfs    445.4M  183.1M   262.3M  41% /
devtmpfs       106.8M     4.0K   106.8M   0% /dev
tmpfs          114.9M     0    114.9M   0% /dev/shm
tmpfs          114.9M  320.0K   114.6M   0% /run
tmpfs          114.9M     0    114.9M   0% /sys/fs/cgroup
tmpfs          114.9M     0    114.9M   0% /tmp
tmpfs          114.9M     0    114.9M   0% /media/ram
tmpfs          114.9M     0    114.9M   0% /var/volatile
tmpfs          23.0M     0    23.0M   0% /run/user/0
root@colibri-vf:~# top_
```

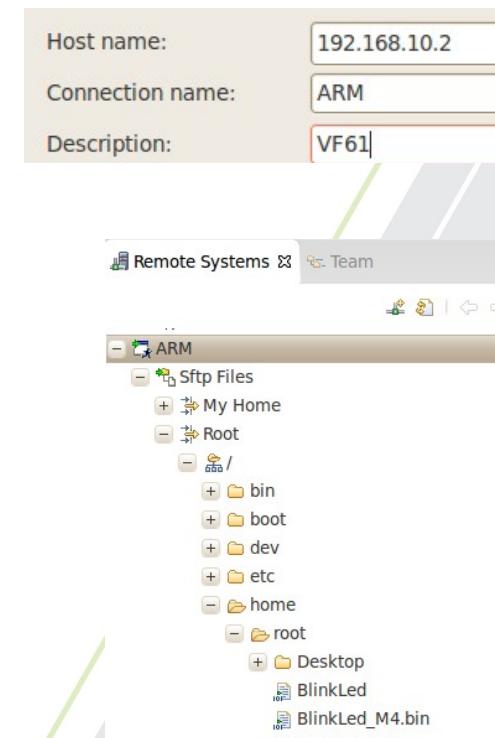
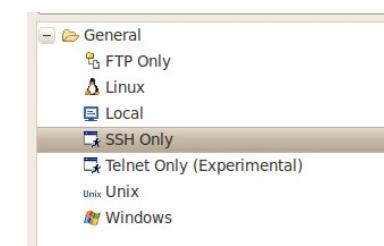


Configurando o Eclipse

Cross-Compiler Linux A5

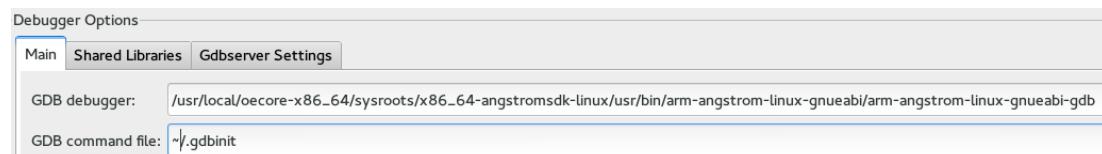
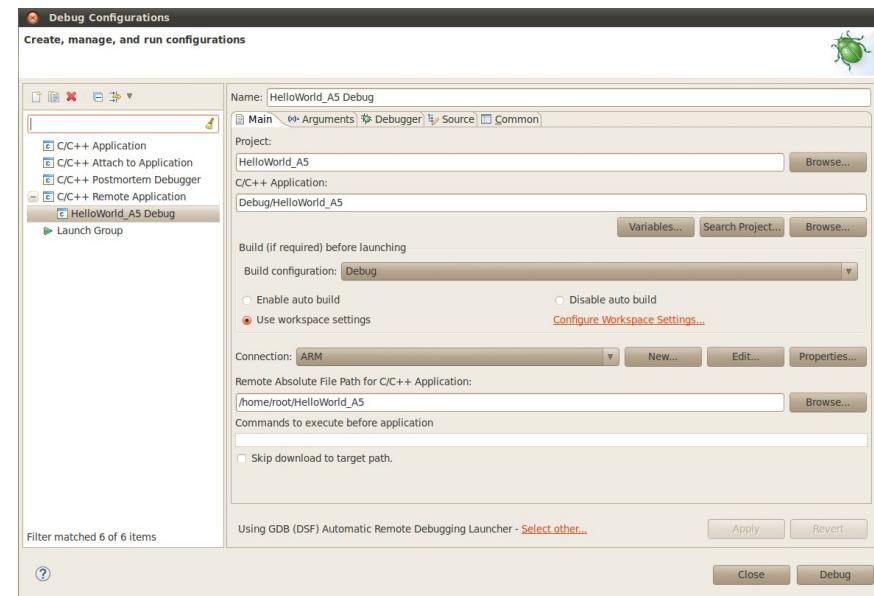
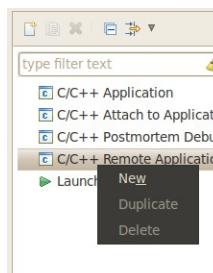


Conectando a placa via SSH



Debug no Eclipse

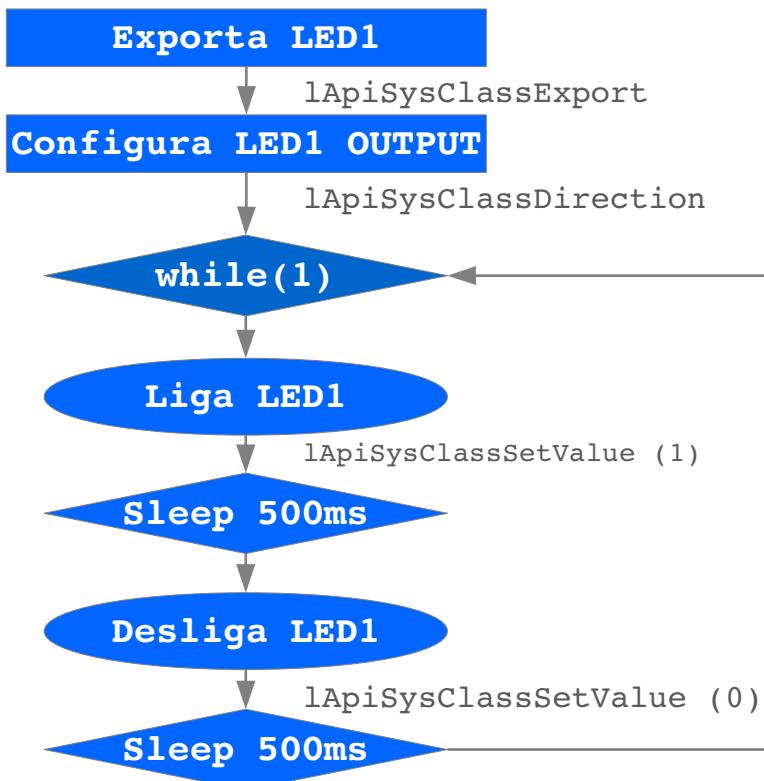
Debug Configurations



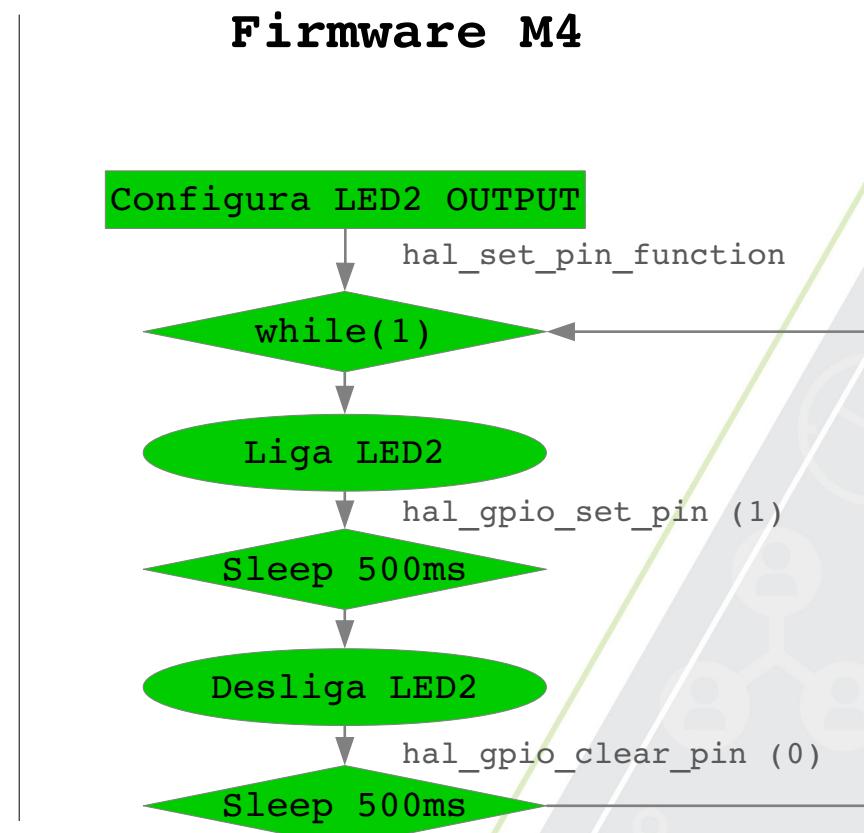
```
$ echo "set sysroot remote:/" > ~/.gdbinit
```

Exemplo 1 - BlinkLed

LINUX



Exemplo 2 - BlinkLed_M4

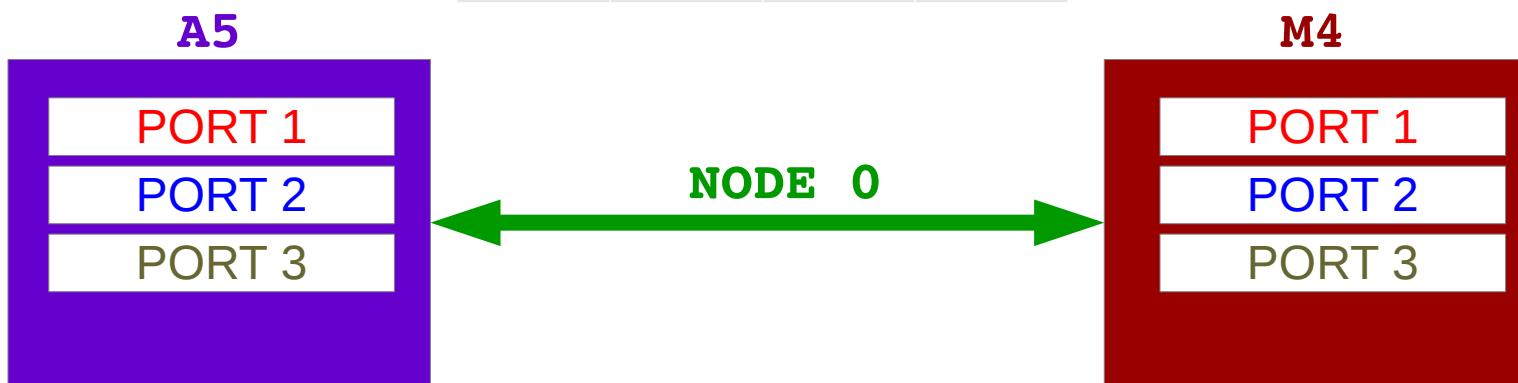


MultiCore Communication library

- Inicialização/Informação
 - **mcc_initialize** (Inicializa MCC para um determinado NODE)
 - **mcc_destroy** (Destroi MCC para determinado NODE)
 - **mcc_get_info** (Retorna informações sobre o MCC)
- Gerenciamento de EndPoint
 - **mcc_create_endpoint** (Cria um endpoint)
 - **mcc_destroy_endpoint** (Destroi um endpoint)
- Comunicação
 - **mcc_send** (Envia mensagem para determinado endpoint)
 - **mcc_receive_copy/mcc_receive_nocopy** (Recebe mensagem de um endpoint)
 - **mcc_msgs_available** (Retorna a quantidade de mensagens na fila do endpoint)
 - **mcc_free_buffer** (Limpa o buffer da última mensagem da função **mcc_receive_nocopy**)

MultiCore Communication library

	Core	Node	Port
A5	0	0	1
M4	1	0	2



```
mcc_initialize(0);
mcc_create_endpoint([0,0,1],1);
mcc_send([1,0,2])
mcc_recv_copy([0,0,1])
```

```
mcc_initialize(0);
mcc_create_endpoint([1,0,2],2);
mcc_send([0,0,1])
mcc_recv_copy([1,0,2])
```

Exemplo 3 - MCC

LINUX

Conf LED1+BOTA01

Inicializa MCC

while(1)

RECEBE MSG

receive_msg

if(BOTA02 == 1)

Desliga LED1

Liga LED1

if(BOTA01 == 1)

ENVIA BT1 (0)

ENVIA BT1 (1)

send_msg

Firmware M4

Conf LED2+BOTA02

Inicializa MCC

while(1)

RECEBE MSG

mcc_recv_copy

if(BOTA01 == 1)

Desliga LED2

Liga LED2

if(BOTA02 == 1)

ENVIA BT2 (0)

ENVIA BT2 (1)

mcc_send

mcc_initialize
mcc_create_endpoint

mcc_initialize
mcc_create_endpoint

Dúvidas?

Raul Rosetto Muñoz
raul.munoz@toradex.com

Toradex Brasil
Rua Luiz Spiandoreli Neto N° 60 – Ed. Paineiras, Sala
304 | Valinhos - SP | Brazil | T: +55 19 3327 3738

Obrigado!