



# Histórico Projetos

Engenharia de Sistemas Embarcados

Prof. Abel Guilhermino

Tópico: Histórico de Projetos da disciplina ESE



# Histórico de Projetos

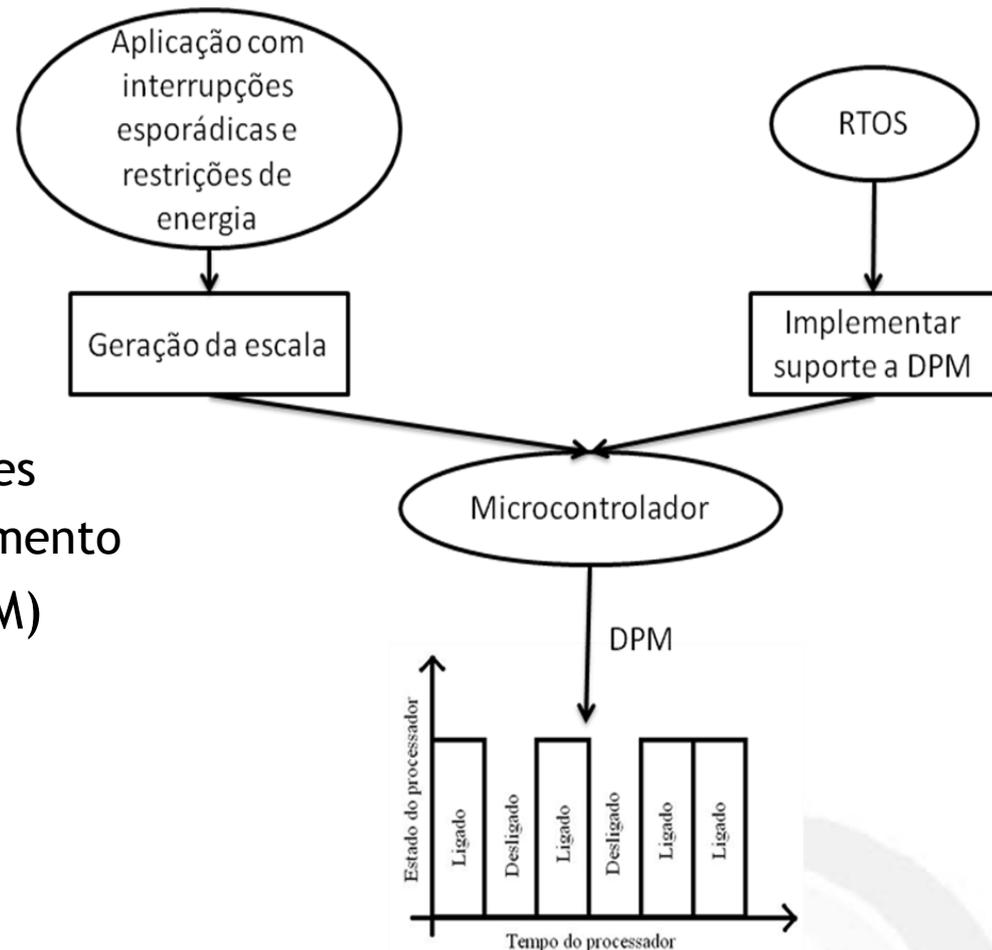
- 2009.1
  - Energy-Aware Hard Real-Time Embedded Systems
  - Avaliação de Consumo de Energia no ARM7
  - Método de Inspeção Visual Automática em Linha de Produção Industrial usando o ARM7
- 2010.1
  - Arquitetura de Medição de Consumo de Energia para Aplicações baseadas em FPGA
  - Análise de Tempo Máximo e Energia de uma Tarefa nos modos THUMB e ARM para o ARM7
  - Redução de Erro em Geo Localização utilizando Fusão de Sensores
- 2011.1
  - Ajuste Dinâmico da Frequência da CPU Baseado no Tráfego da Rede Ethernet
  - Compactação de Imagens usando ARM7

# Histórico de Projetos

- 2012.1
  - Estudo de Consumo de Energia de Aplicativos usando a Plataforma Beagleboard e o Sistema Operacional Android
  - Análise de Consumo de Energia de Arquivos de Áudio no ARM com Android
  - Avaliação de Consumo de Energia do Algoritmo Reverse Time Migration (RTM) em GPU
  - Biblioteca de Suporte a Aplicações “Energy Harvest Aware”
  - Framework Arduino

# 1. Energy-Aware Hard Real-Time Embedded Systems

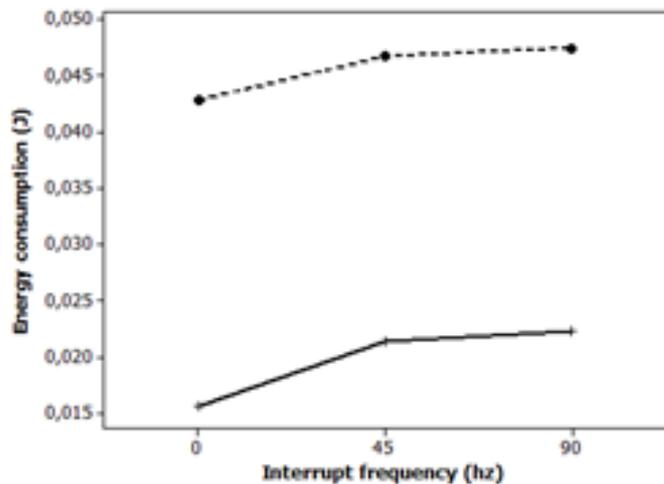
- Equipe
  - Bruno Nogueira
  - Bruno Silva
  - Daniele Santos
  - Gustavo Melo
- Escalonamento Híbrido
  - Tratamento Seguro de Interrupções
  - RTOS para implementar escalonamento
- Dynamic Power Management (DPM)



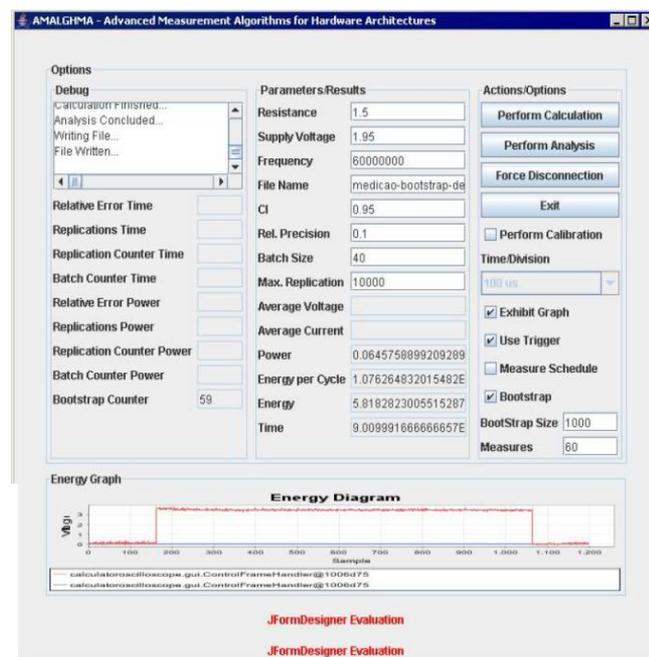
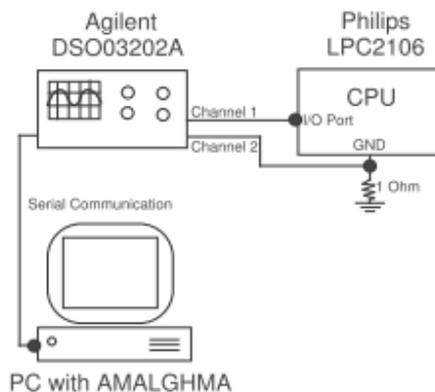
# 1. Energy-Aware Hard Real-Time Embedded Systems

- Escalonamento
  - Tempo de Resposta Tarefa ( $R_i$ ) [Melo]
  - Esquema de Medição (Amalghma) [Maciel]
  - ARM7TDMI, LPC2106
- Resultados
  - Ganho de 135% quando usa DPM

$$R_i = \sum_{n=1}^i c_{ch,n} + \sum_{\forall p \in ic(R_i)} \sum_{m=1}^{nofTask(p)} c_{p,m} + \sum_{\forall interrupt} \left[ \frac{R_i}{T_{interrupt}} \right] c_{interrupt}$$

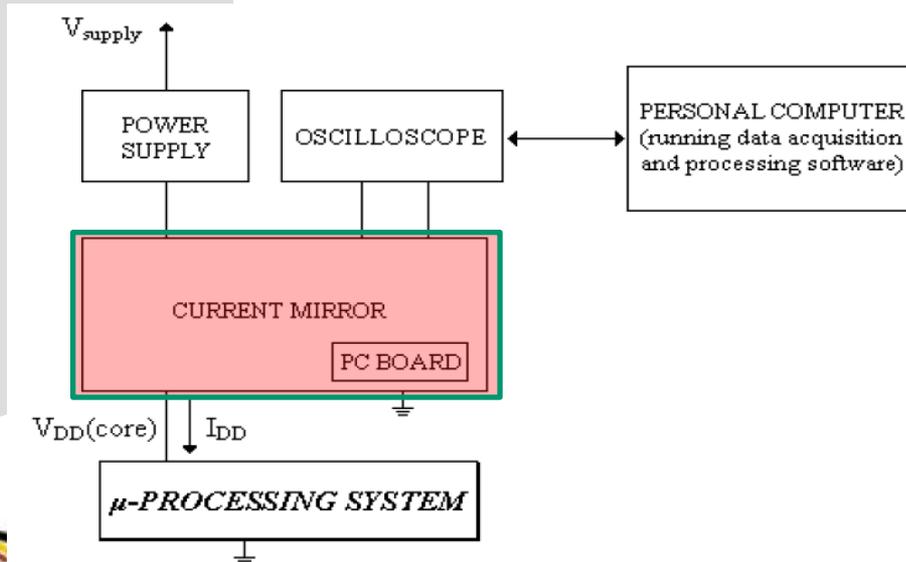


(a)

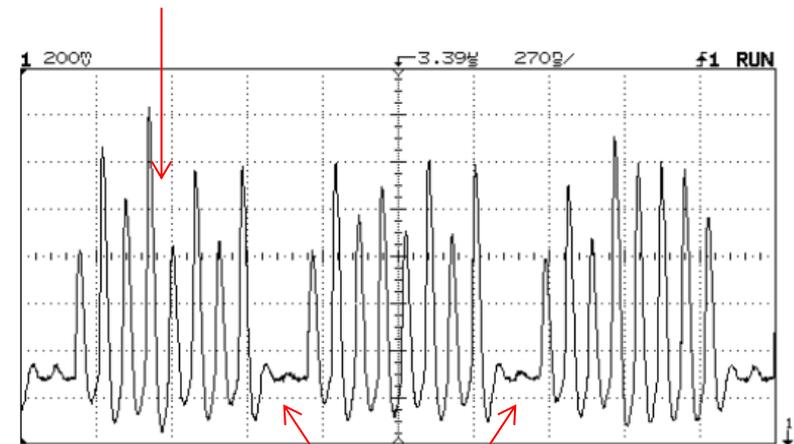


## 2. Avaliação de Consumo de Energia no ARM7

- Equipe
  - Abner Barros
  - Thiago Figueiredo
  - Max Santana
  - Luis Eugênio
- Caracterização de Instruções do ARM7



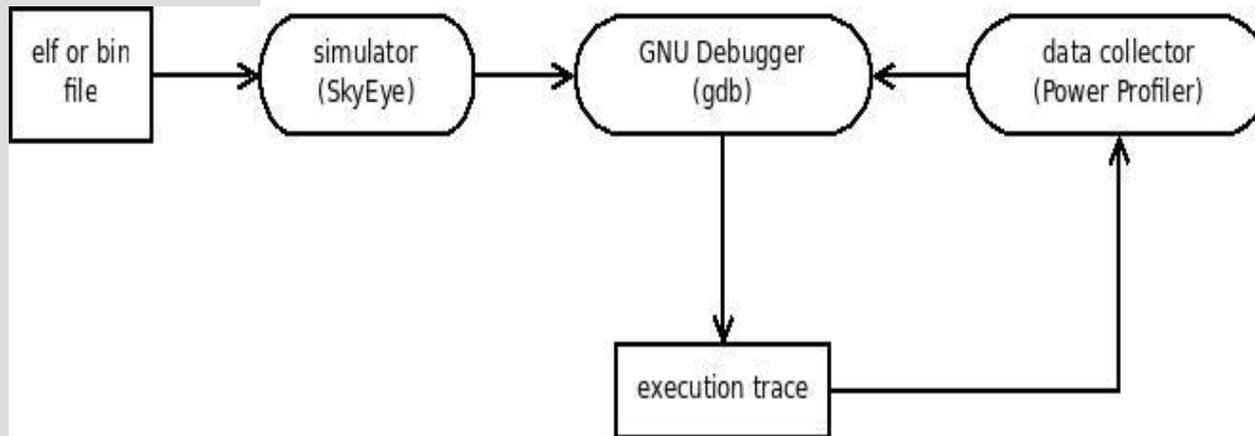
### Energia do ciclo



**Break do depurador**

## 2. Avaliação de Consumo de Energia no ARM7

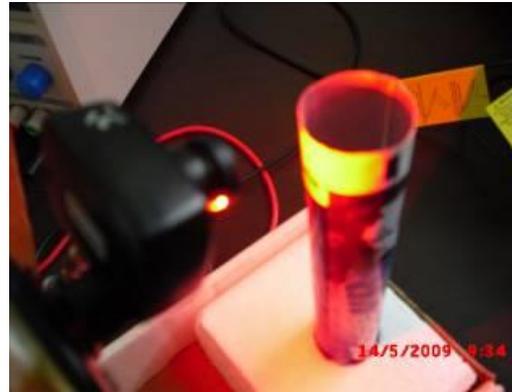
- Framework



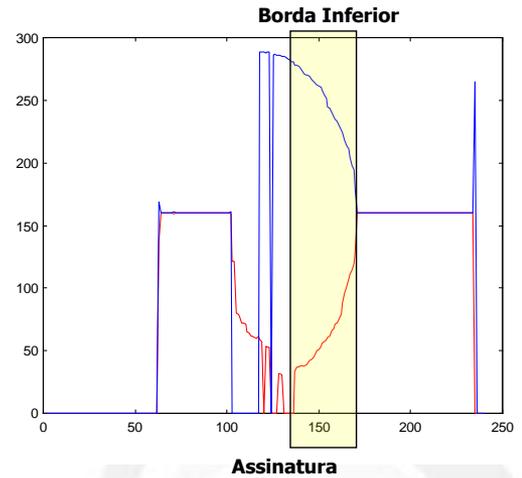
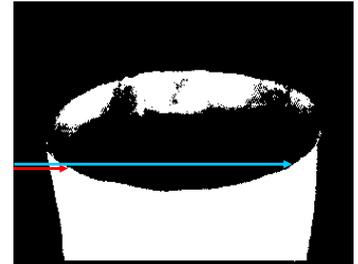
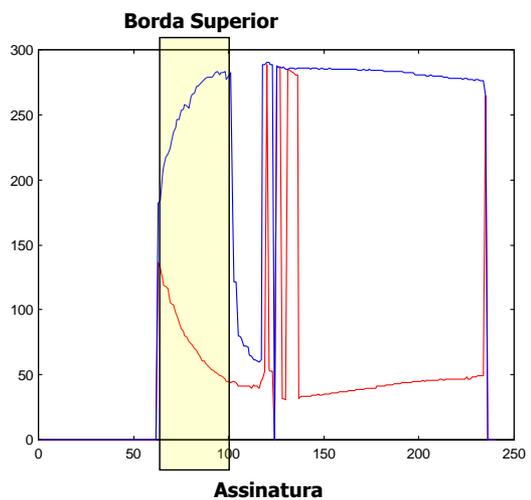
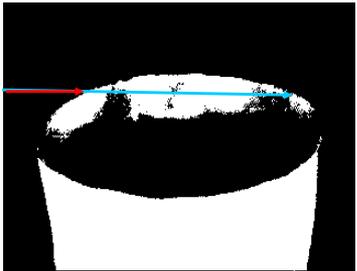
Auto./Industrial	Consumer	Office	Network	Security	Telecomm.
basicmath	jpeg	ghostscript	dijkstra	blowfish enc.	CRC32
bitcount	lame	ispell	patricia	blowfish dec.	FFT
qsort	mad	rsynth	{CRC32}	pgp sign	IFFT
susan (edges)	tiff2bw	sphinx	(sha)	pgp verify	ADPCM enc.
susan (corners)	tiff2rgba	stringsearch	(blowfish)	rijndael enc.	ADPCM dec.
susan (smoothing)	tiffdither			rijndael dec.	GSM enc.
	tiffmedian			sha	GSM dec.
	typeset				

### 3. Método de Inspeção Visual Automática em Linha de Produção Industrial usando o ARM7

- Equipe
  - Ygo
  - Rafael Macieira
  - Ricardo Lins
- Detecção de tubos ruins
- Câmera CCD monocromática
- Uso de LED para ajudar na segmentação
- Segmentação via hardware
  - Comparadores analógicos para binarização do sinal NTSC da câmera

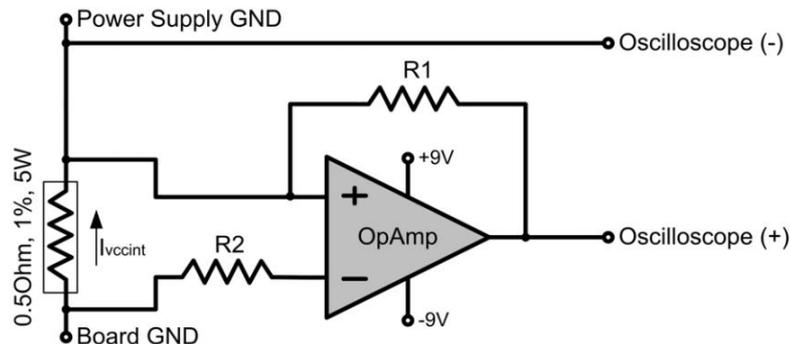
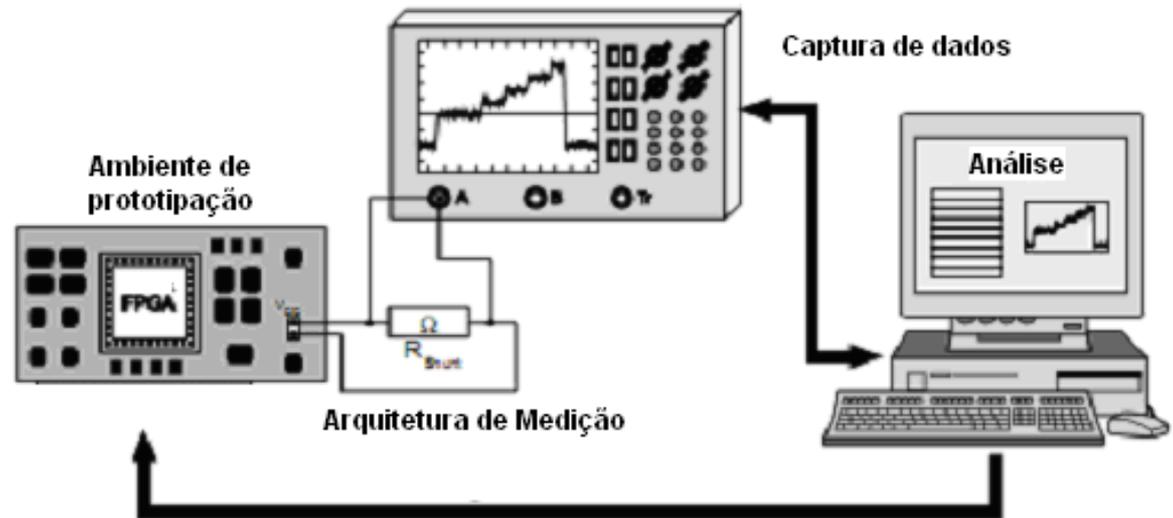


### 3. Método de Inspeção Visual Automática em Linha de Produção Industrial usando o ARM7



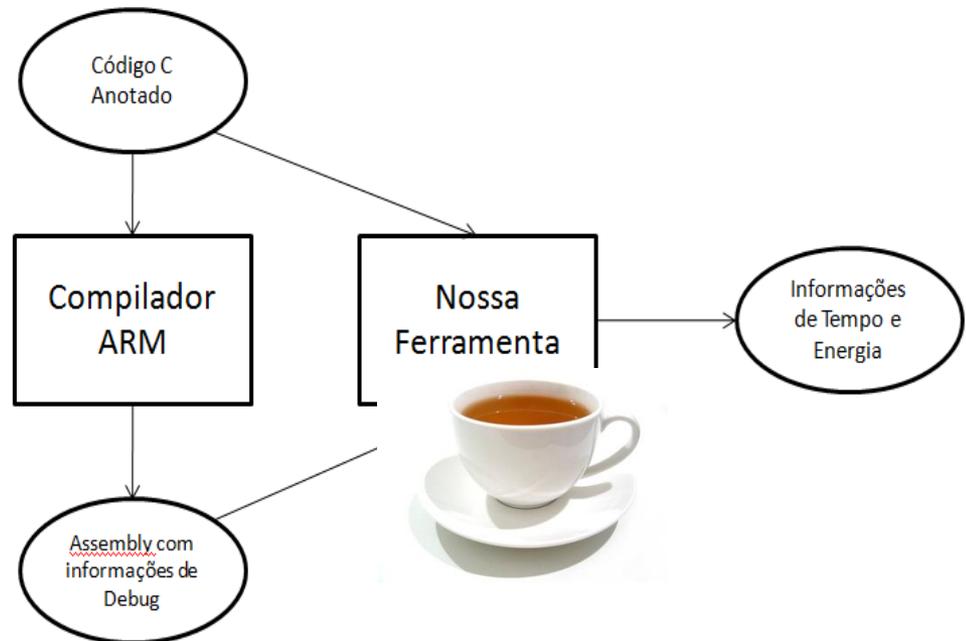
# 4. Arquitetura de Medição de Consumo de Energia para Aplicações baseadas em FPGA

- Equipe
  - Gilliano Ginno
  - Leonardo Nunes
  - Marcelo Santos
  - Viviane Lucy
- Uso de FPGA Xilinx
- Circuito Condicionador



# 5. Análise de Tempo Máximo e Energia de uma Tarefa nos modos THUMB e ARM para o ARM7

- Equipe
  - Bruno Correia
  - Bruno Montenegro
  - Daniel Gomes
  - Samir Oliveira
- Cálculo de tempo de uma tarefa
  - Uso de Gragos
- Programação Linear Inteira (ILP)



# 5. Análise de Tempo Máximo e Energia de uma Tarefa nos modos THUMB e ARM para o ARM7

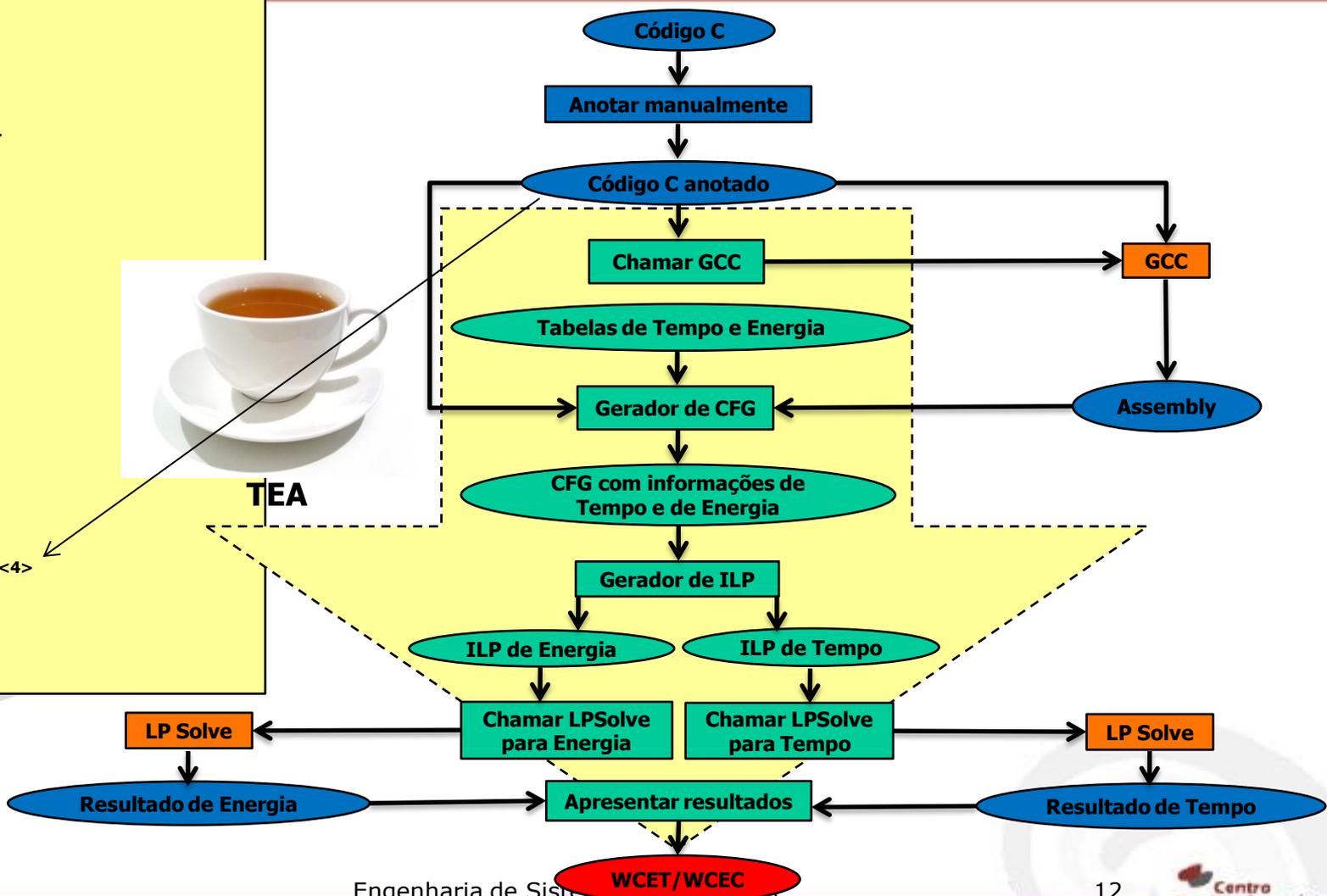
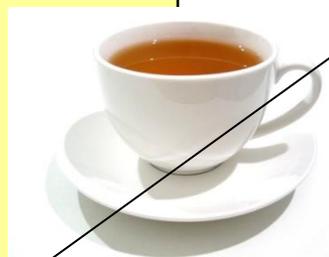
```

#include <stdio.h>
#include <stdlib.h>

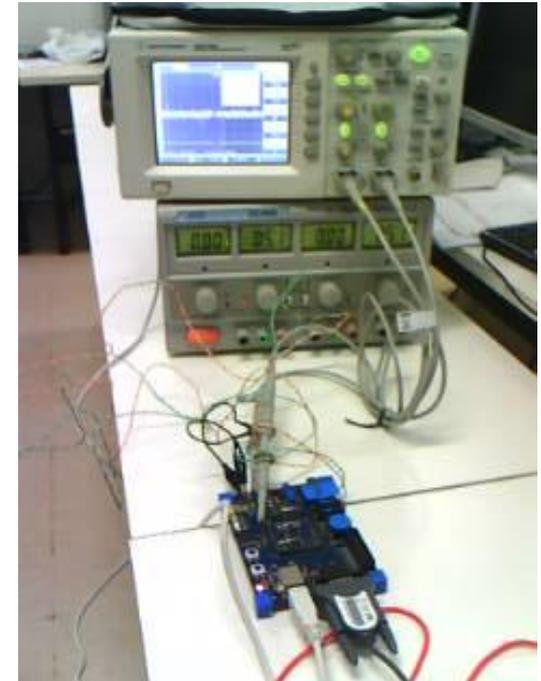
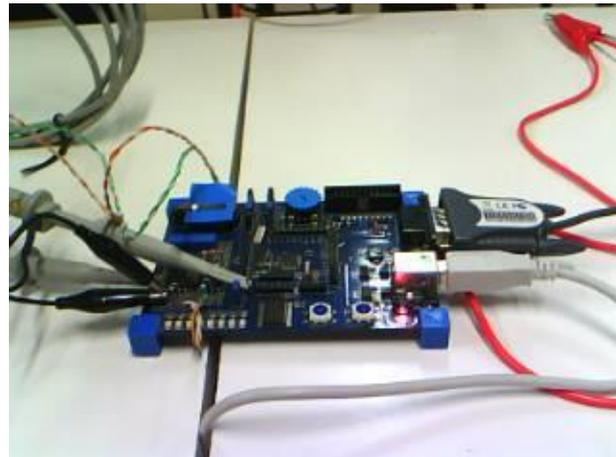
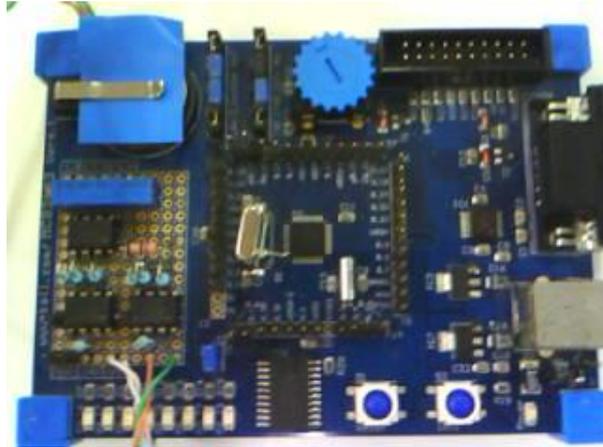
int test(int a, int b)
{
    int c;
    c = a + b;
    if (c < 10)    //<0.33>
    {
        c = 9;
    }
    else
    {
        c = c + 1;
    }
    return c;
}

int somar( int x, int y) {
    return x + y;
}

int main()
{
    int a,b,c,i;
    int d;
    a = 10;
    b = 20;
    c = somar(a,b);
    d = somar(a,b);
    for (i=0;i<4;i++) //<4>
    {
        a = a+b;
        test(a, b);
    }
    return 1;
}
    
```



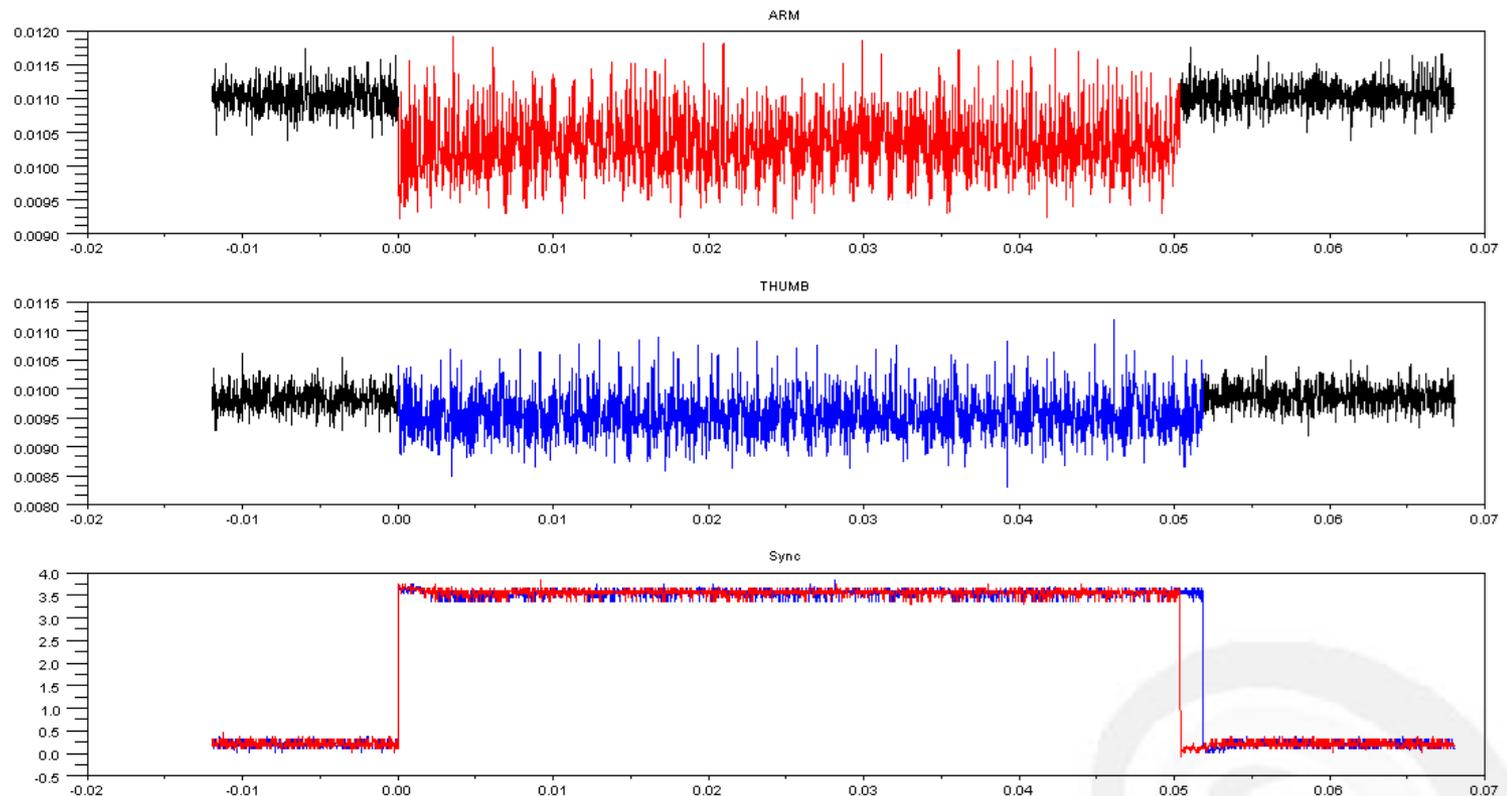
## 5. Análise de Tempo Máximo e Energia de uma Tarefa nos modos THUMB e ARM para o ARM7



# 5. Análise de Tempo Máximo e Energia de uma Tarefa nos modos THUMB e ARM para o ARM7

## Benchmark - Powerstone

Plot do adpcm.c



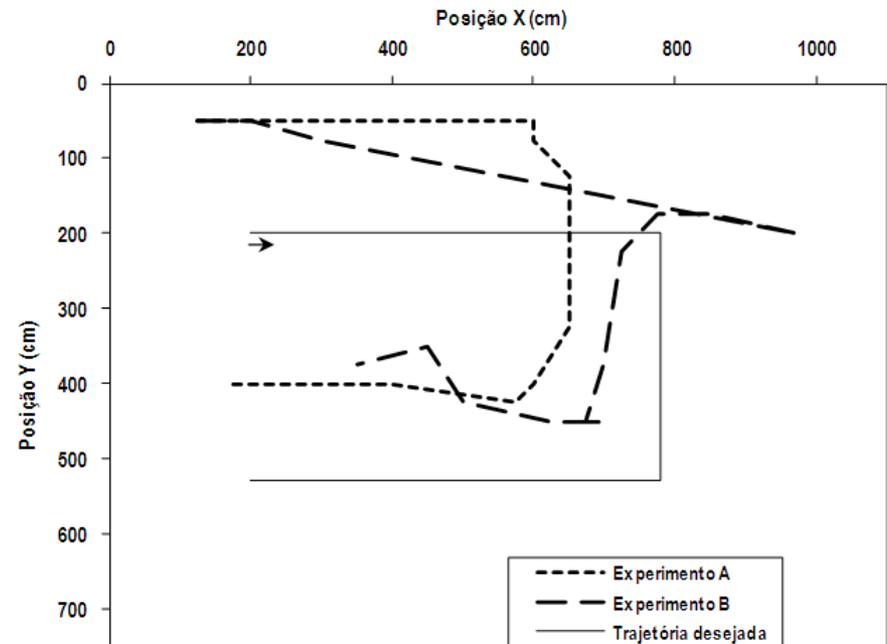
## 5. Análise de Tempo Máximo e Energia de uma Tarefa nos modos THUMB e ARM para o ARM7

Powerstone	Energia Medida (ARM) (nW.s)	Energia Medida (Thumb) (nW.s)	TEA WCEC (ARM) (nW.s)	TEA WCEC (Thumb) (nW.s)
adpcm.c	518927.536226	493914.076990	16.808.978	23.565.944
bcnt.c	21587.013093	21129.892456	219.344	334.282
blit.c	318877.895373	313309.754915	10.769.873	17.525.896
crc.c	21122.248773	24592.924012	9.159.008	13.043.102
des.c	26020.544285	28541.816694	10.790.388	17.124.045
jpeg.c	-----	-----	293.826.948.590	438.435.627.025

Baseando-se nos resultados obtidos com o benchmark powerstone podemos afirmar que o modo ARM é mais eficiente que o modo Thumb no que diz respeito a consumo de energia e tempo de execução.

# 6. Redução de Erro em Geo Localização utilizando Fusão de Sensores

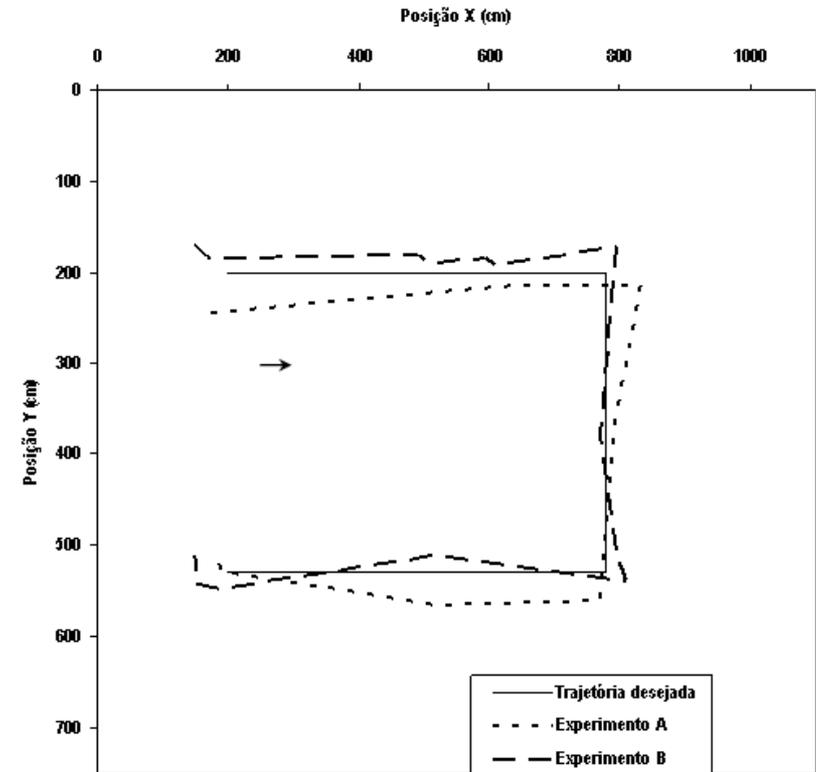
- Equipe
  - Alessandro Glauber
  - Daniel Leite
  - Claudio Oliveira
  - Fábio Leitão
- Uso de Acelerômetro
  - O acelerômetro fornece como saída o valor da aceleração ( $m/s^2$ ) nos três eixos (x,y,z).



**Estimado pelo Zigbee**

# 6. Redução de Erro em Geo Localização utilizando Fusão de Sensores

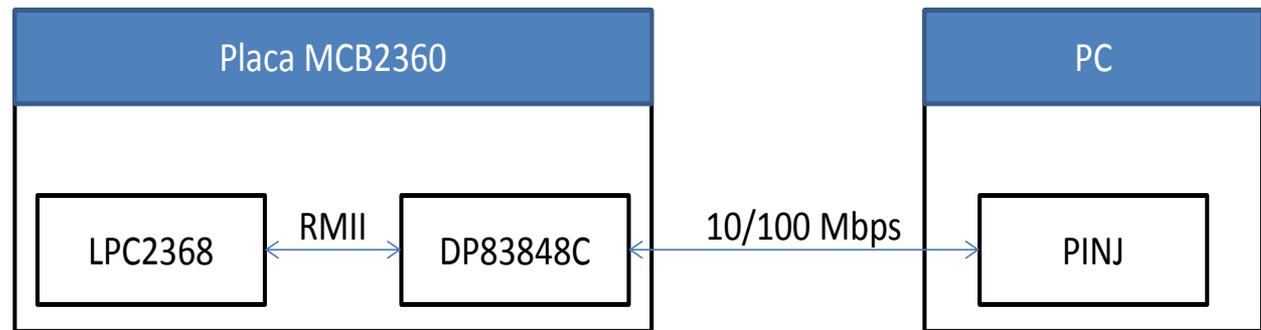
- Fusão (INS + Zigbee)
  - O acelerômetro fornece como saída o valor da aceleração (m/s<sup>2</sup>) nos três eixos (x,y,z).
- Uso de Sensor Zigbee
- Uso de Sensor INS



**Estimado pelo Zigbee/INS**

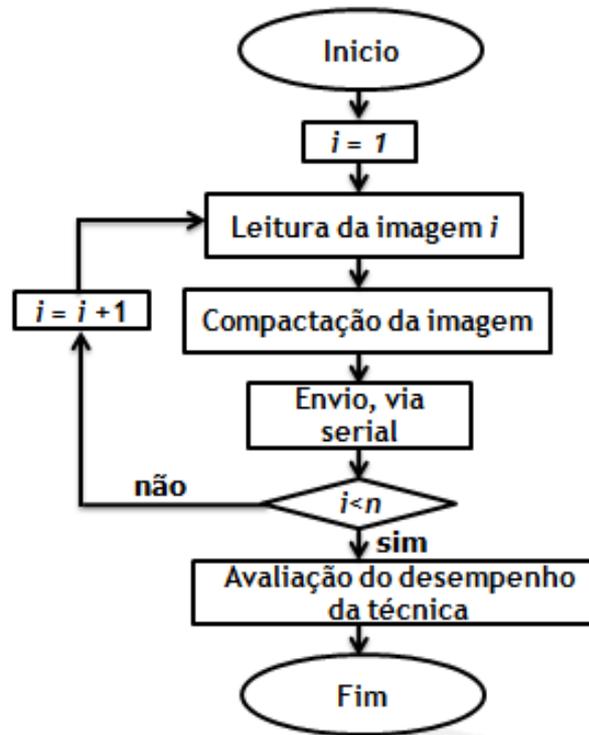
# 7. Ajuste Dinâmico da Frequência da CPU Baseado no Tráfego da Rede Ethernet

- Equipe
  - Diogo Lages
  - Igino Chaves
- Uso do ARM7
- Ethernet
- DFS
- RFE (Reduced Frame Energy)



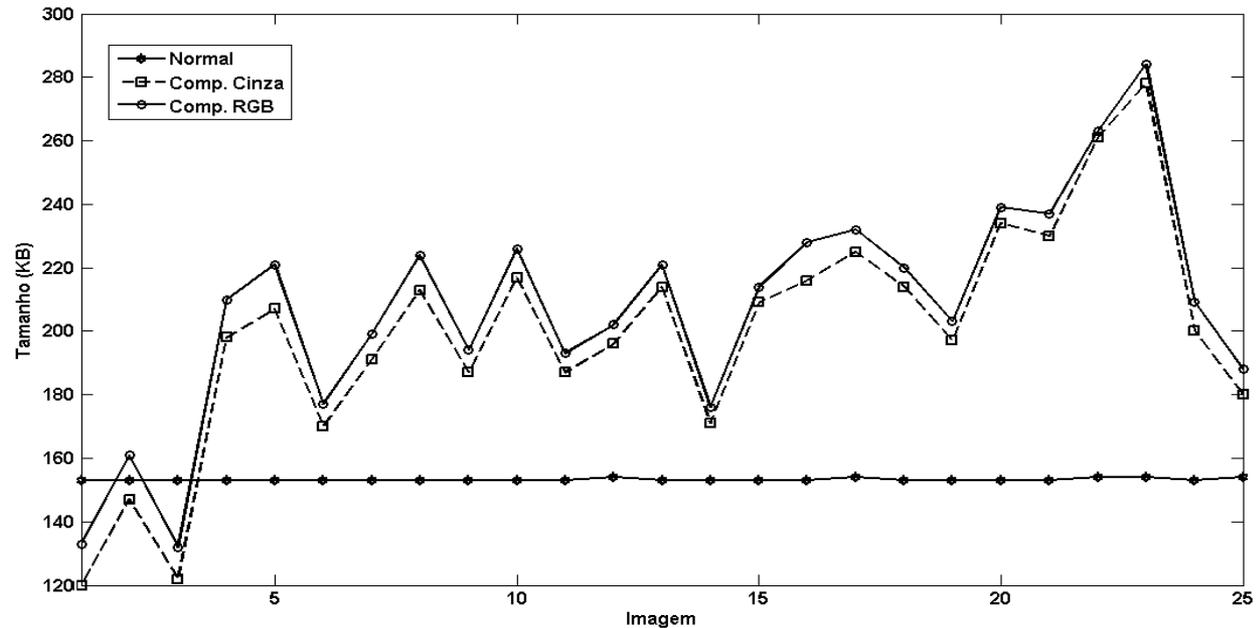
## 8. Compactação de Imagens usando ARM7

- Equipe
  - Sidney Lima
  - Andson Balieiro
- Uso do ARM7
- SDCARD



# 8. Compactação de Imagens usando ARM7

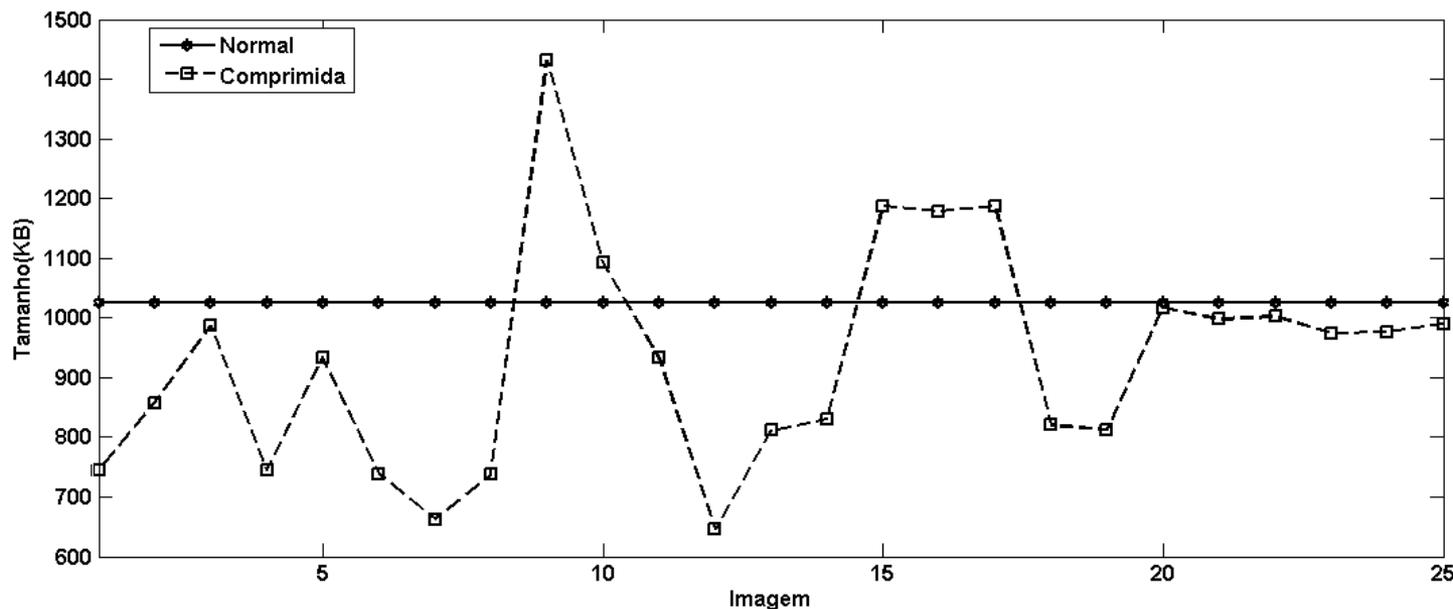
- Imagens Fotográficas



Escola	Tamanho (KB)	Tempo Compressão(s)	Razão de compressão
Cinza	153,2	2,48	1,3
RGB	153,2	2,6	1,35

## 8. Compactação de Imagens usando ARM7

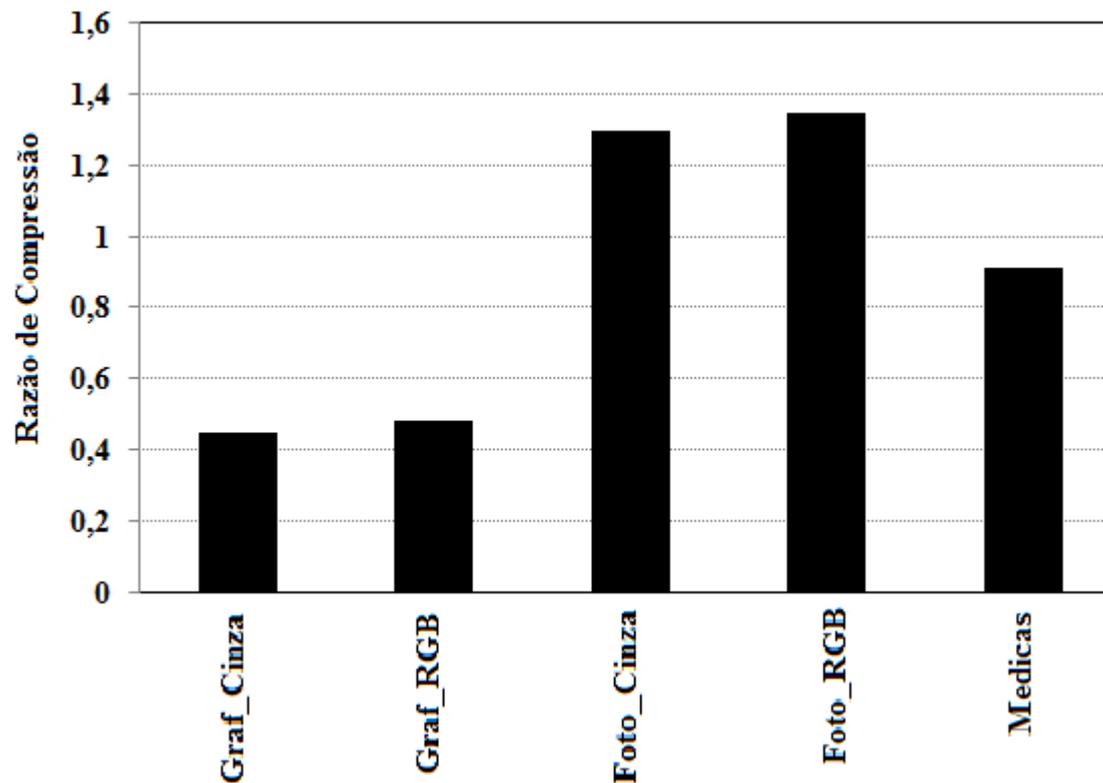
- Imagens Médicas



Tamanho (KB)	Tempo Compressão(s)	Razão de compressão
1026	12,96	0,91

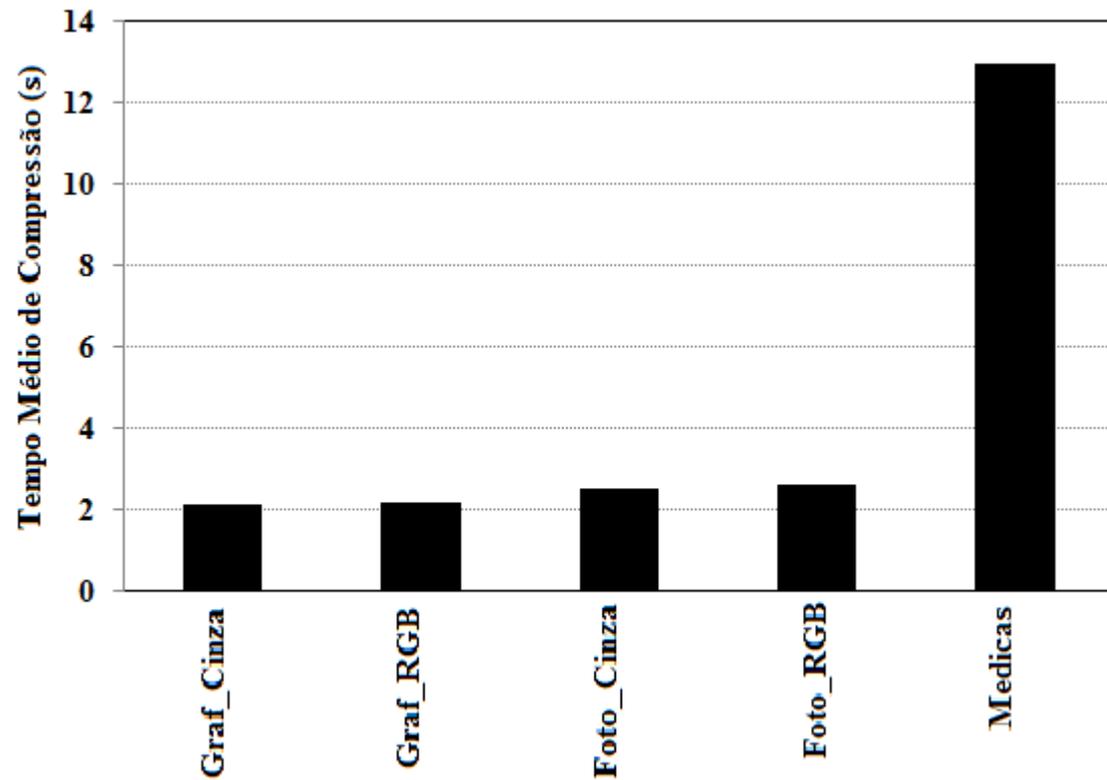
## 8. Compactação de Imagens usando ARM7

- Razão de Compressão



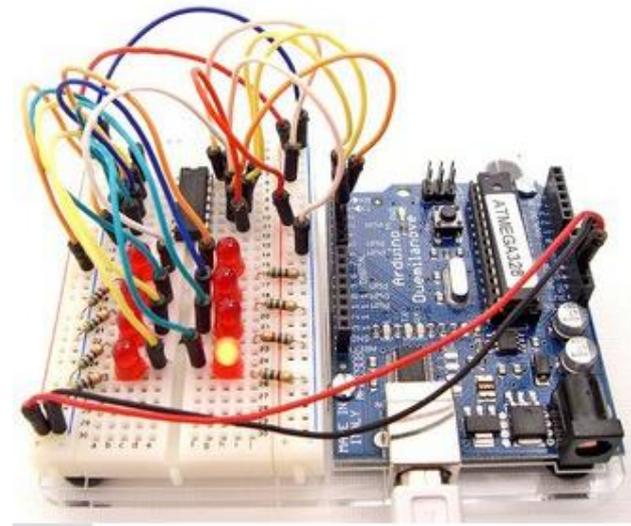
## 8. Compactação de Imagens usando ARM7

- Tempo de Compressão



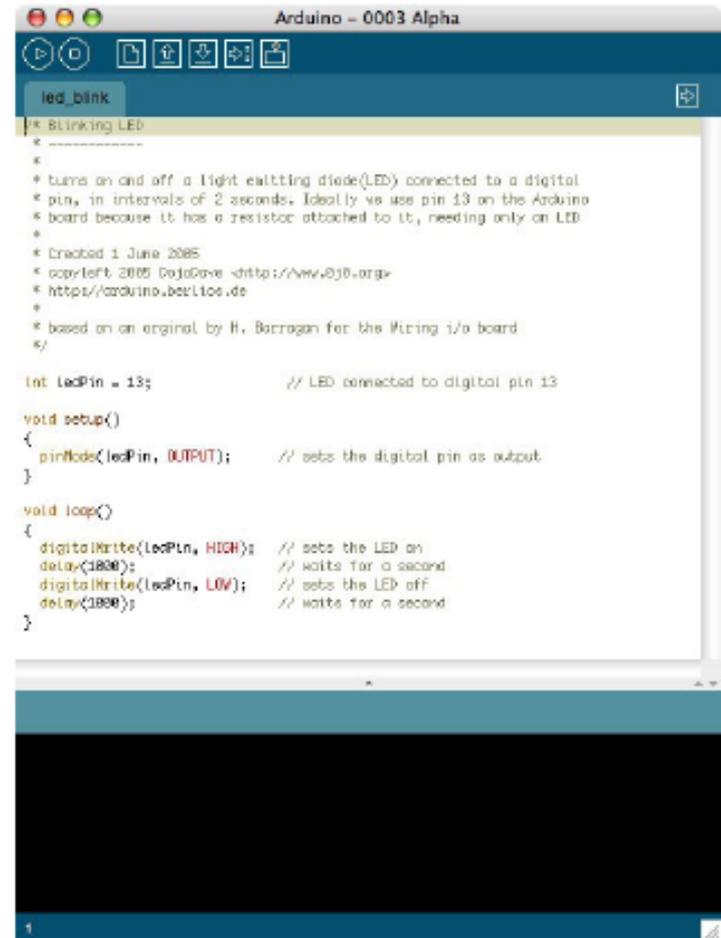
## 9. tAMARINO

- Equipe
    - Ricardo Brasileiro
    - David Ribeiro
    - Giovane Ribeiro
    - David Fraga
  - Uso do Arduino
    - Modelo de Prototipagem de Computação com Arduino
- Toolkit para em Rápida Física com



# 9. tAMARINO

- Problema
  - A IDE não é atrativa para usuários não-especialistas (designers, artistas, entusiastas, pesquisadores)



```
Arduino - 0003 Alpha
led_blink
/* Blinking LED
 *
 * turns on and off a light emitting diode(LED) connected to a digital
 * pin, in intervals of 2 seconds. Ideally we use pin 13 on the Arduino
 * board because it has a resistor attached to it, needing only an LED
 *
 * Created 1 June 2005
 * copyright 2005 David Cuatrecasas <http://www.arduino.org>
 * http://arduino.berlios.de
 *
 * based on an original by H. Barragan for the Wiring i/o board
 */

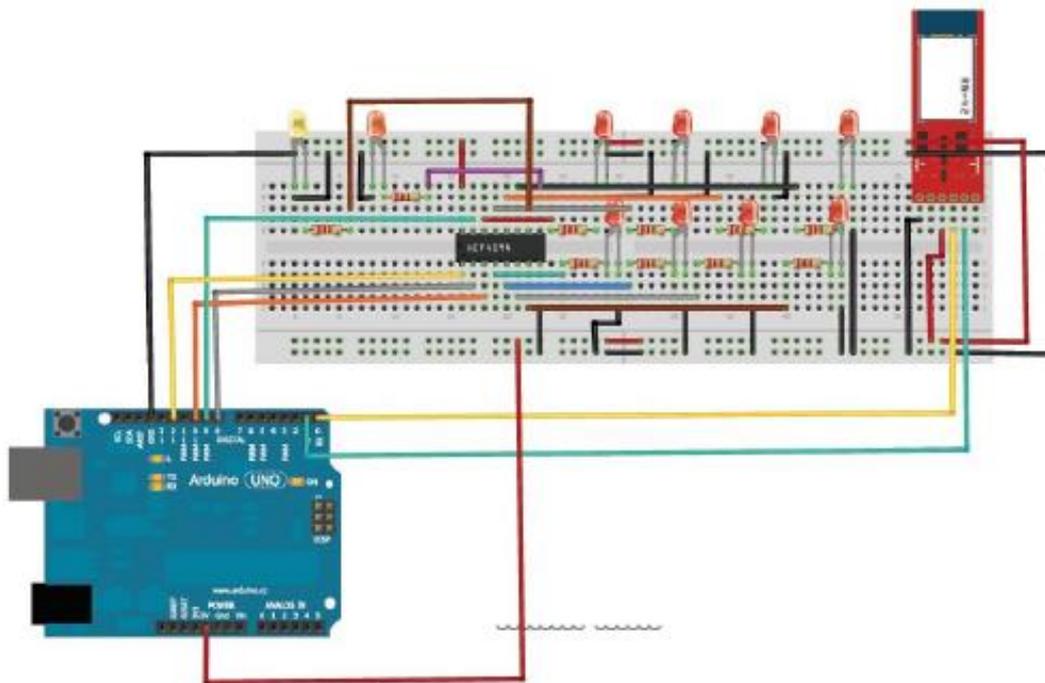
int ledPin = 13;           // LED connected to digital pin 13

void setup()
{
  pinMode(ledPin, OUTPUT); // sets the digital pin as output
}

void loop()
{
  digitalWrite(ledPin, HIGH); // sets the LED on
  delay(1000);                // waits for a second
  digitalWrite(ledPin, LOW);  // sets the LED off
  delay(1000);                // waits for a second
}
```

## 9. tAMARINO

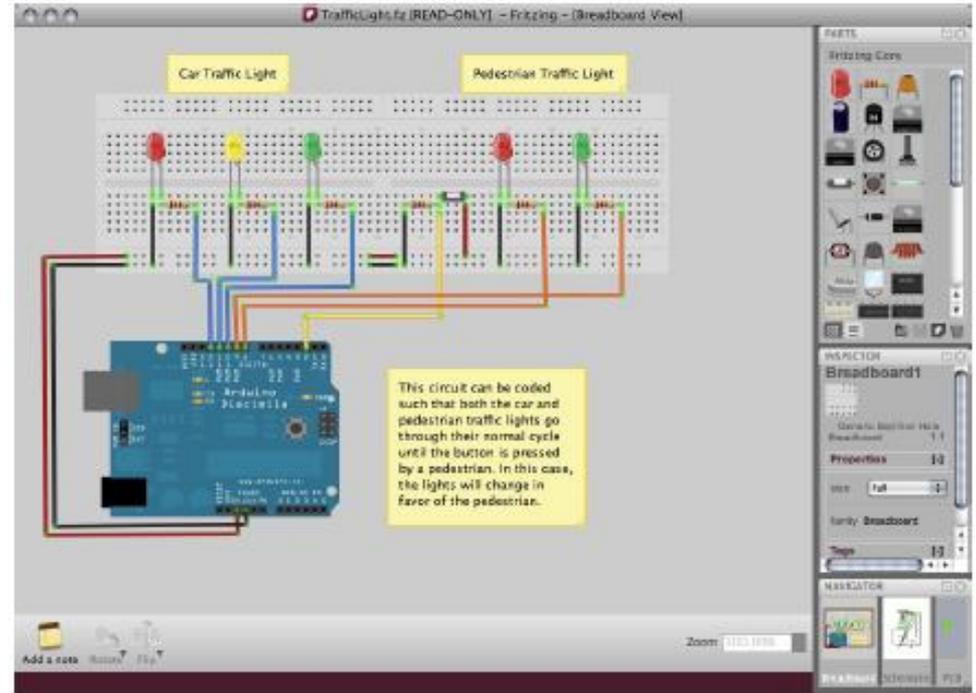
- Problema
  - A prototipagem eletrônica é uma barreira para usuários que não possuem conhecimentos básicos em física e eletrônica.





# 9. tAMARINO

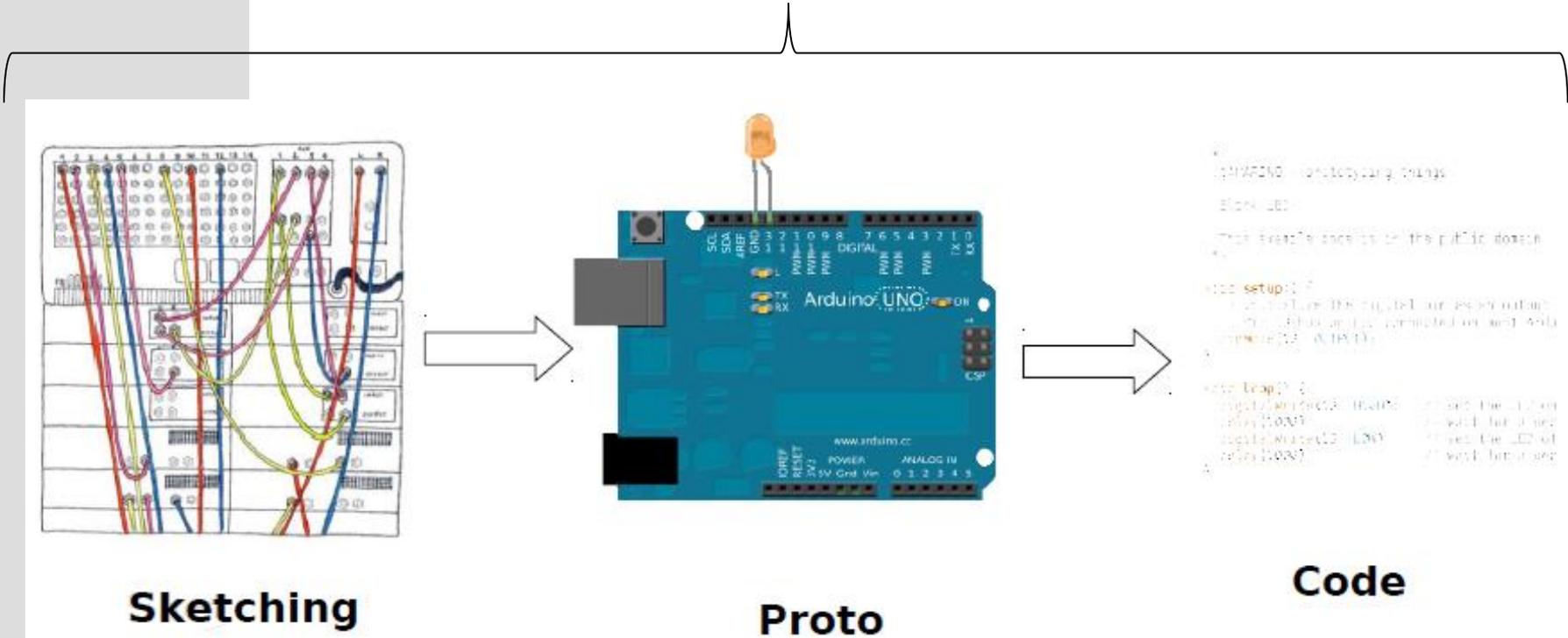
- Problema
  - O Fritzing facilita a documentação mas não é atrativo para usuários leigos em eletrônica.



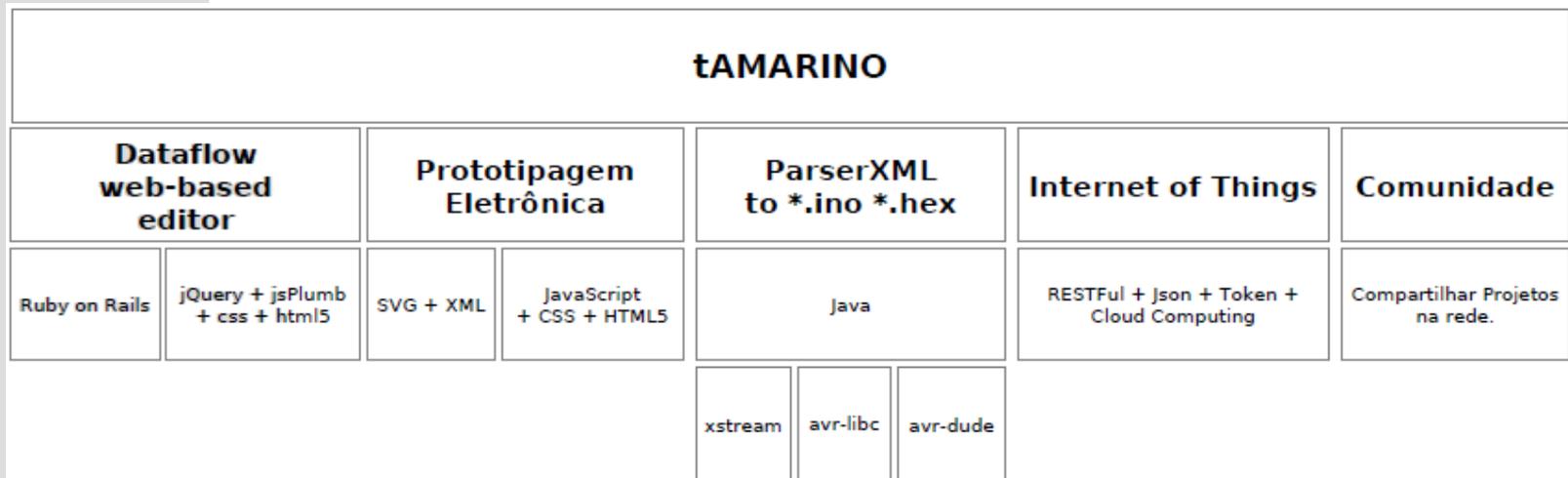
# 9. tAMARINO

- Solução Proposta

## tAMARINO



# 9. tAMARINO



# 9. tAMARINO



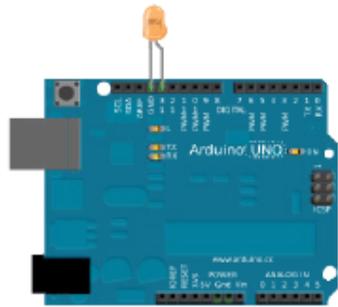
**tCasca**  
Sketch  
Dataflow web-based

tCaule  
dataflow to XML

**XML**

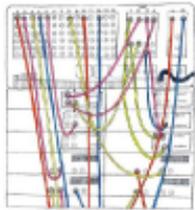
tFruto  
XML to \*.ino

tFolha  
dataflow to SVG

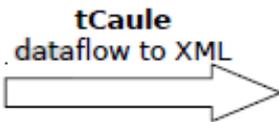


```
/*  
 * tAMARINO - prototyping things  
 *  
 * tFruto - LED  
 *  
 * This example code is in the public domain.  
 */  
  
void setup() {  
  // initialize the digital pin as an output.  
  // Pin 13 has an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}  
  
void loop() {  
  digitalWrite(13, HIGH); // set the LED on  
  delay(1000);           // wait for a second  
  digitalWrite(13, LOW); // set the LED off  
  delay(1000);          // wait for a second  
}
```

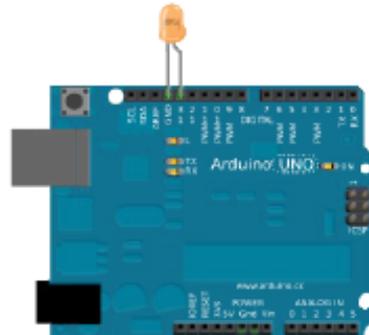
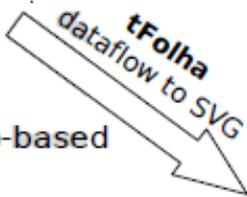
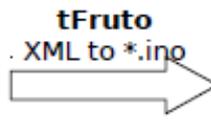
# 9. tAMARINO



**tCasca**  
Sketch  
Dataflow web-based



**XML**

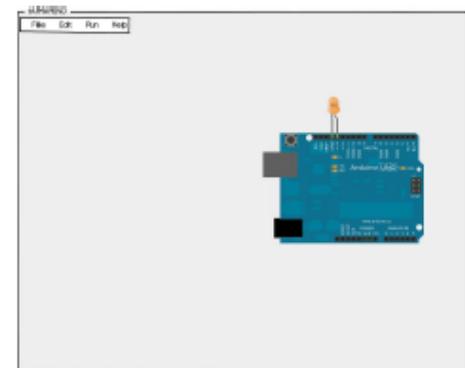


```
...  
tAMARINO - prototyping things  
...  
tFruto - LED  
...  
This example code is in the public domain.  
...  
//  
  
void setup() {  
  // initialize the digital pin as an output.  
  // For more on this see an LED connected on most Arduino boards:  
  pinMode(13, OUTPUT);  
}  
...  
  
void loop() {  
  digitalWrite(13, HIGH); // set the LED on  
  delay(1000);           // wait for a second  
  digitalWrite(13, LOW); // set the LED off  
  delay(1000);          // wait for a second  
}
```

# 9. tAMARINO

The screenshot shows the tAMARINO software interface. At the top, there is a menu bar with 'File', 'Edit', 'Run', and 'Help'. Below the menu bar, there are three tabs: 'Sketch', 'Proto', and 'Code'. On the left side, there are two main component lists: 'Input' and 'Output'. The 'Input' list includes POTENTIOMETER, PUSHBUTTON, TOGGLE, PHOTOCCELL, INFRARED, ACCELEROMETER, RFID, and TEMPAUMID. The 'Output' list includes LED, DISPLAY, LCD, PIEZO, LOUDSPEAK, DC MOTOR, SERVO, STEPPER, SOLENOID, and RELAY. A 'Functions' sub-menu is open, showing options like BLINK, BUTTON, FADE, SMOOTH, and STATE. In the center workspace, there are two circuit components: a 'BLINK' component with two delay values of 1000 and a 'PORT' component with a value of 13. On the right side, there are three more menus: 'Control' (BANG, METRO, CHANGE, DELAY, COUNTER, TRIGGER, SELECT) and 'Operators' (+, -, \*, /, %, ==, !=, >=, <=).

This screenshot shows the 'Run' dialog box in the tAMARINO interface. The dialog has two buttons: 'Pause' and 'Code'. Below the dialog, a circuit diagram is visible, showing a component labeled 'BLINK' connected to a component labeled 'LED'. The 'BLINK' component has two delay values of 1000.



```

tAMARINO
File Edit Run Help

/*
 * Example - prototyping phase
 * Blink LED
 * This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards
  pinMode(13, OUTPUT);
}

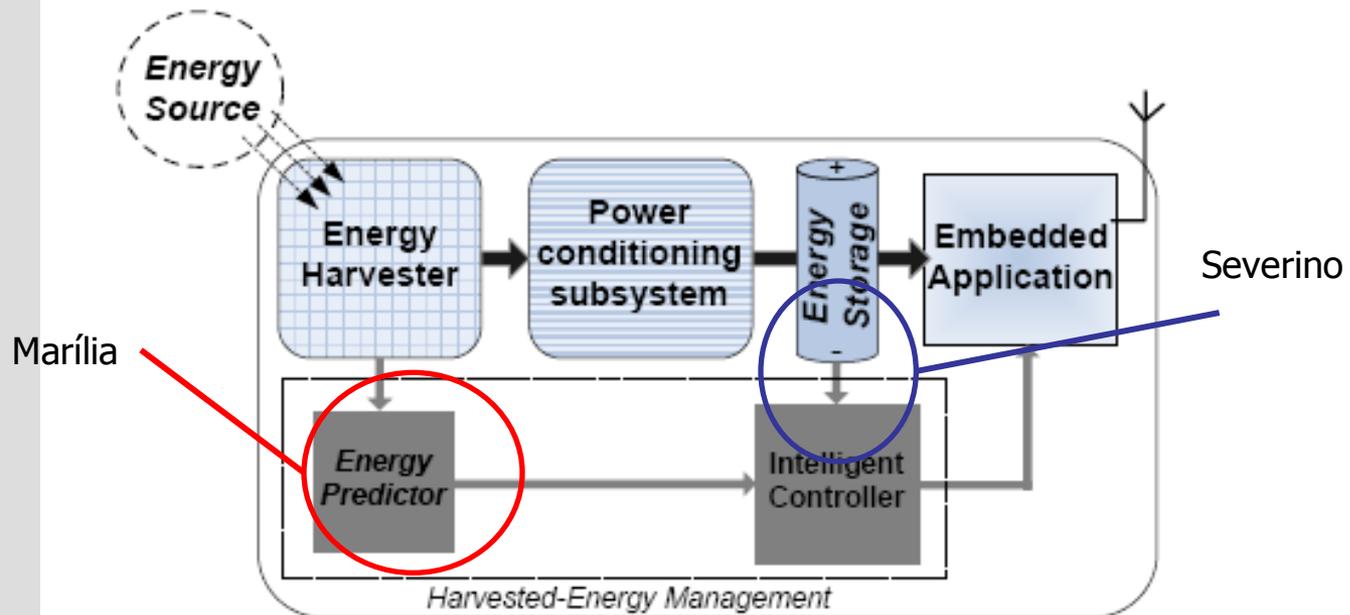
void loop() {
  digitalWrite(13, HIGH); // set the LED on
  delay(1000);            // wait for a second
  digitalWrite(13, LOW);  // set the LED off
  delay(1000);            // wait for a second
}

```

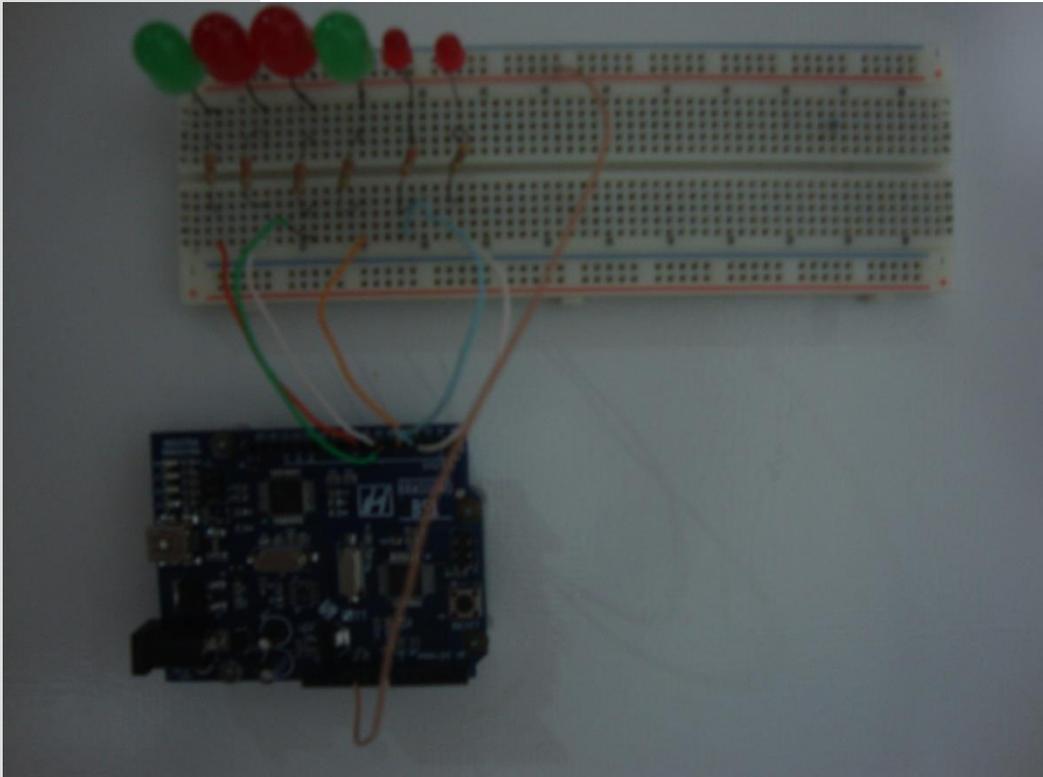
## 10. Energy Harvest Aware

- Equipe
  - Marília
  - Severino
- Energy Harvest Aware
  - Economia de Energia

# 10. Energy Harvest Aware



## 10. Energy Harvest Aware



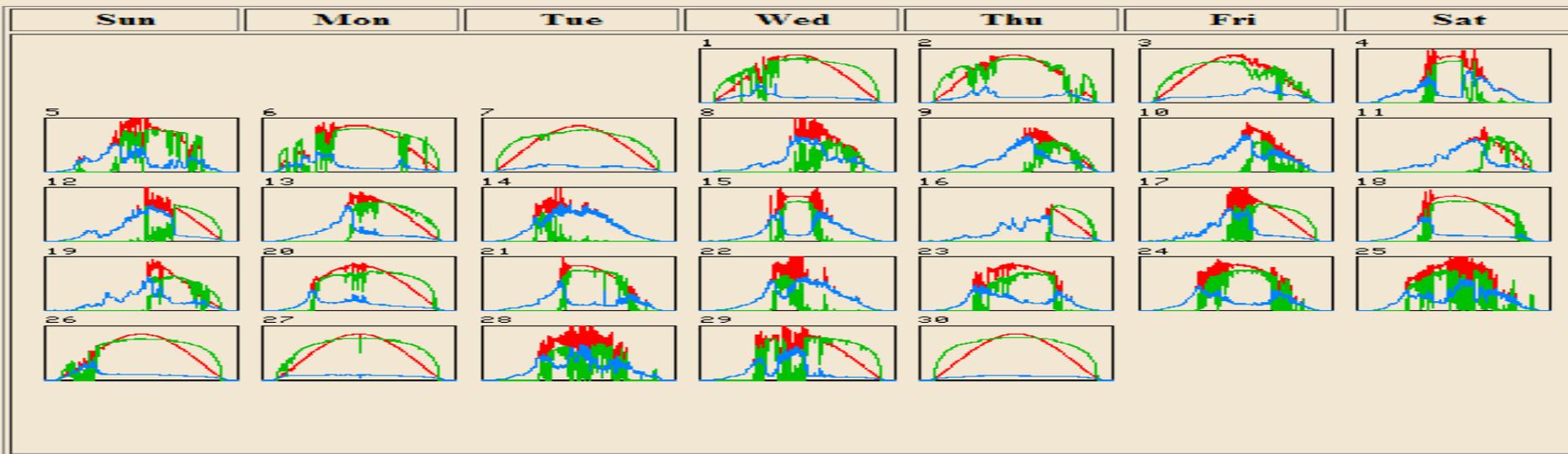
- Objetivo: manter a lâmpada sempre acesa: regular a tensão de acordo com energia disponível
  - Manter a maior iluminação possível a cada intervalo de 1 hora.
- Fontes de energia: bateria e painel solar

# 10. Energy Harvest Aware

- Energia (hipotético)
  - Bateria dura até 24h com um led aceso
  - Consumo aplicação: 240W/h
- Algoritmo Predição Energia Solar
  - Interface em arquivo
  - Período dados históricos: 09 a 27/06/2011, UCLA
  - Versão atual: acende os leds de acordo com previsão de energia
- Algoritmo Estado de Carga da Bateria
  - Incluir na próxima versão

# 10. Energy Harvest Aware

June 2011 Solar Calendar



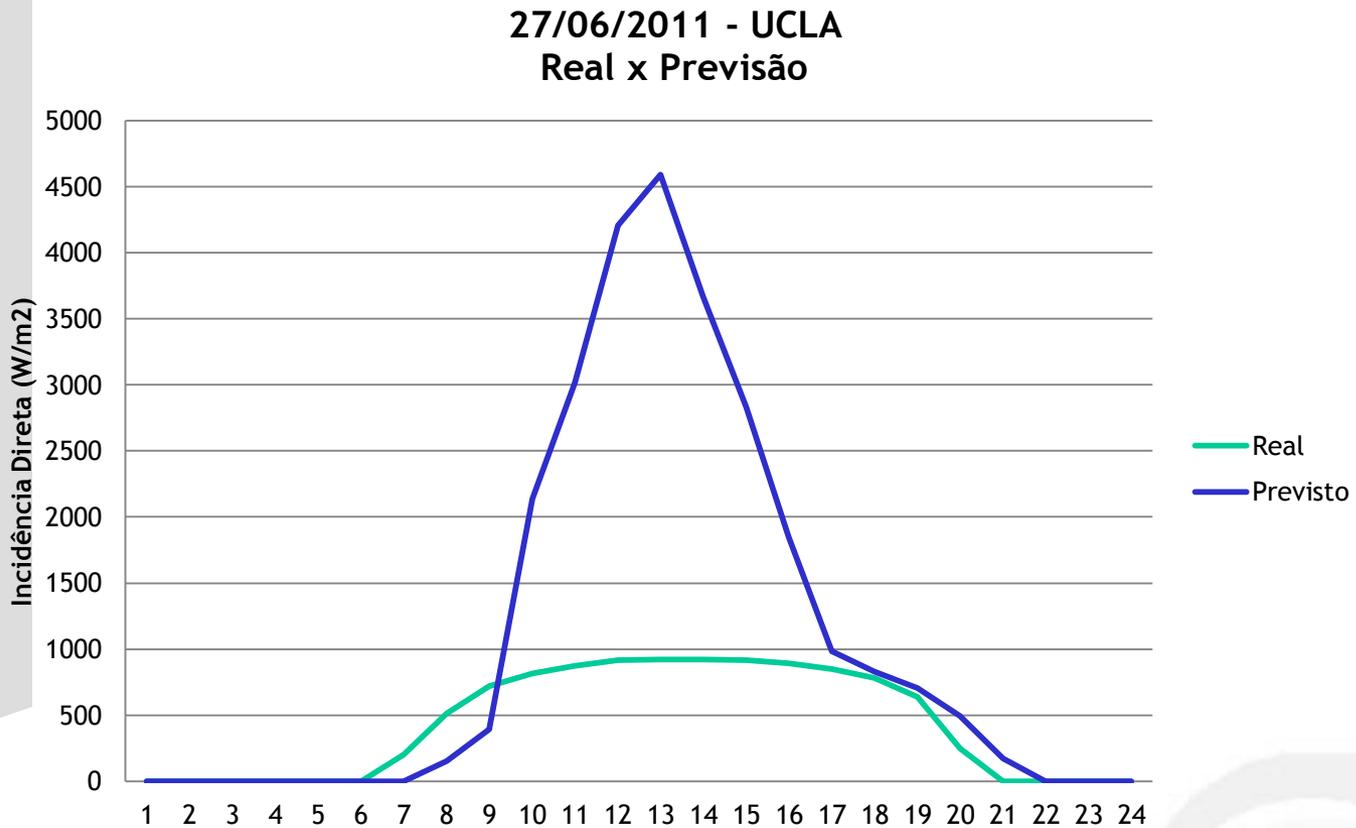
[Previous Month](#)

Red = Global, Green = Direct, Blue = Diffuse

[Next Month](#)

Parâmetro	Cenário Ideal [2]	Implementado Arduino UNO
D - numero dias anteriores	20	18
N - numero de slots num dia	48 (intervalos 30 min)	24 (intervalos 1h)
K - numero slots anteriores	7	7

# 10. Energy Harvest Aware



# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

- Equipe
  - Millena
  - Maria Cireno
  - Maryane Chagas
- Uso Android
  - Estimativa de Energia
  - Formatos: mp3, ogg, wav
- Uso da Beagleboard
- Estilos Musicais
  - Instrumental
  - RAP
  - R&B
  - Rock

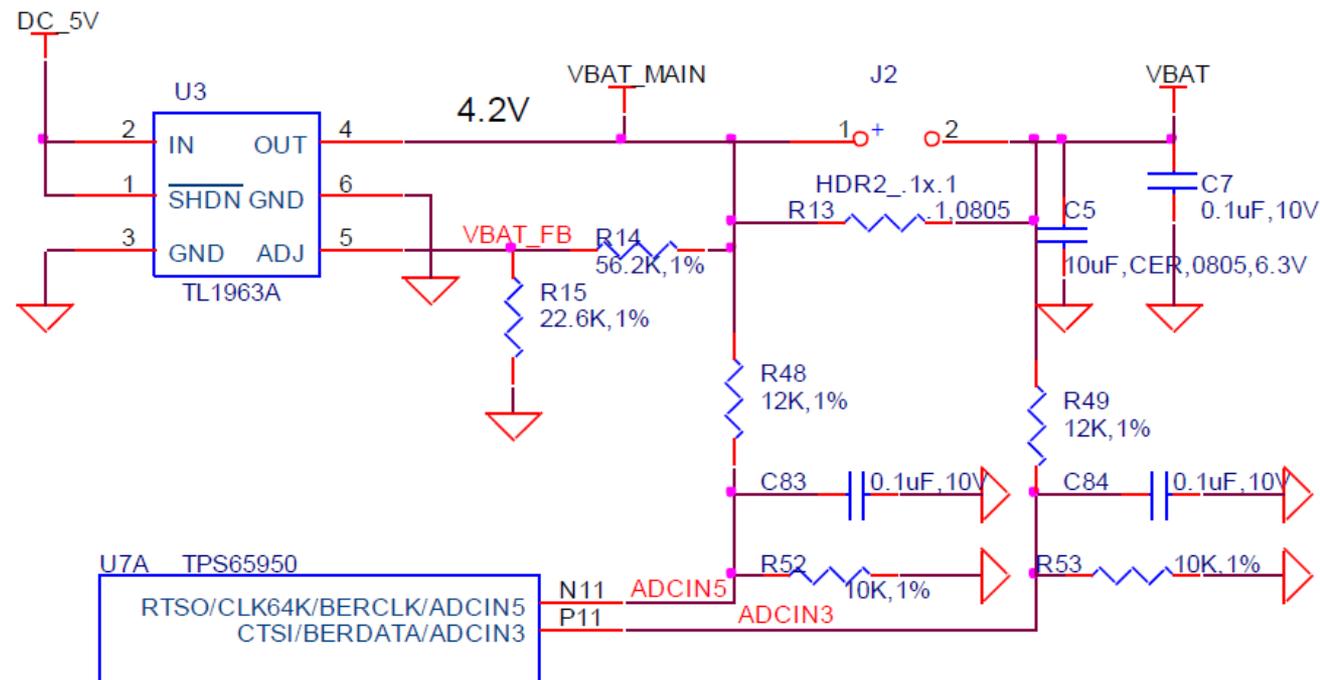
# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

- BeagleBoard-xM Rev-C
  - Processador ARM DM3730CBP com frequência de 1GHz;
  - Saída de áudio;
  - MicroSD;
  - USB;
  - TPS65950 Power Management;
  - I2C Communication.



# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

- BeagleBoard-xM Rev-C
  - Corrente calculada através da diferença de potencial no resistor R13 (0,10hms)
  - Diferença entre os valores de ADCIN 3 e ADCIN5



# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

- Medição do consumo
  - I2C tools para acessar o barramento I2C;
  - Acesso ao TPS65950 através da I2C;
  - Configuração de registradores para habilitar medição do consumo - ativação do clock do MADC, ativação do MADC, seleção dos canais 3 e 5;
  - Leitura do consumo via leitura de registradores - acesso aos valores do ADCIN3 e ADCIN5 de saída do TPS65950.

# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

- Android 4.0.3
- Shell Script
- Android SDK - API15
- ARM EABI v7a
- Ferramentas:
  - Eclipse - desenvolvimento Java para Android



# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

- Shell script para obtenção dos valores de tensão em ACDIN3 e ADCIN5 durante execução de uma música;
- Estes serão os dados de entrada para o cálculo da energia pela aplicação;
- Estas informações não podem ser obtidas diretamente pela Aplicação Android (Java).

```
# !/bin/bash
```

```
i2cget -f -y 1 0x49 0x91 b  
i2cset -f -y 1 0x49 0x91 0x90 b  
i2cset -f -y 1 0x49 0x91 0x00 b  
i2cset -f -y 1 0x49 0x91 0x90 b  
i2cset -f -y 1 0x4a 0x00 0x01 b  
i2cset -f -y 1 0x48 0xbb 0x08 b  
i2cset -f -y 1 0x4a 0x06 0x28 b  
i2cset -f -y 1 0x4a 0x07 0x00 b  
i2cset -f -y 1 0x4a 0x08 0x28 b  
i2cset -f -y 1 0x4a 0x09 0x00 b
```

```
i=1  
while [  $$( ( 1 < 2 ) ) = 1$  ]  
do  
hora=$(date +%T)  
echo "Amostra: $i Hora: $hora"  
i="expr $i + 1"
```

```
i2cset -f -y 1 0x4a 0x12 0x20 b  
i2cget -f -y 1 0x4a 0x3d w  
i2cget -f -y 1 0x4a 0x41 w
```

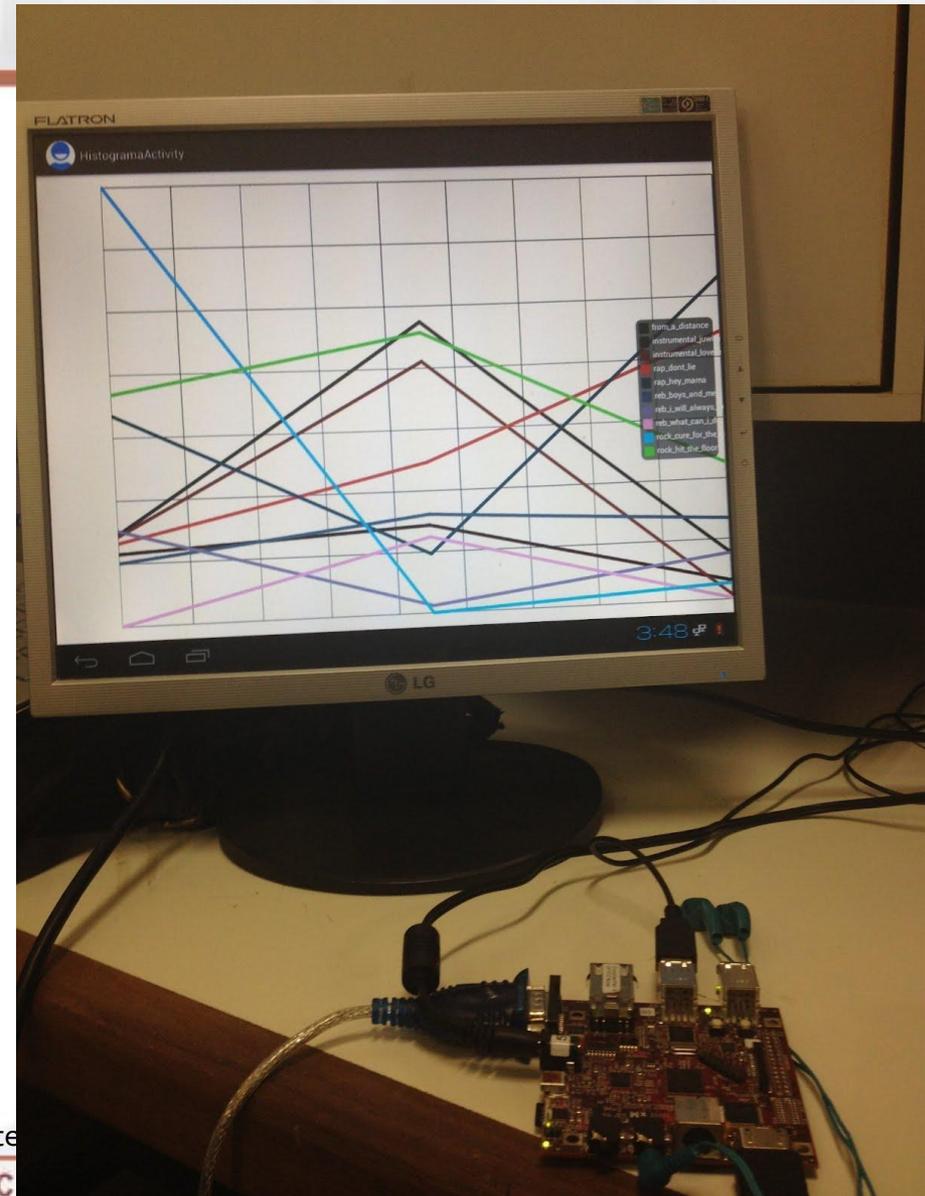
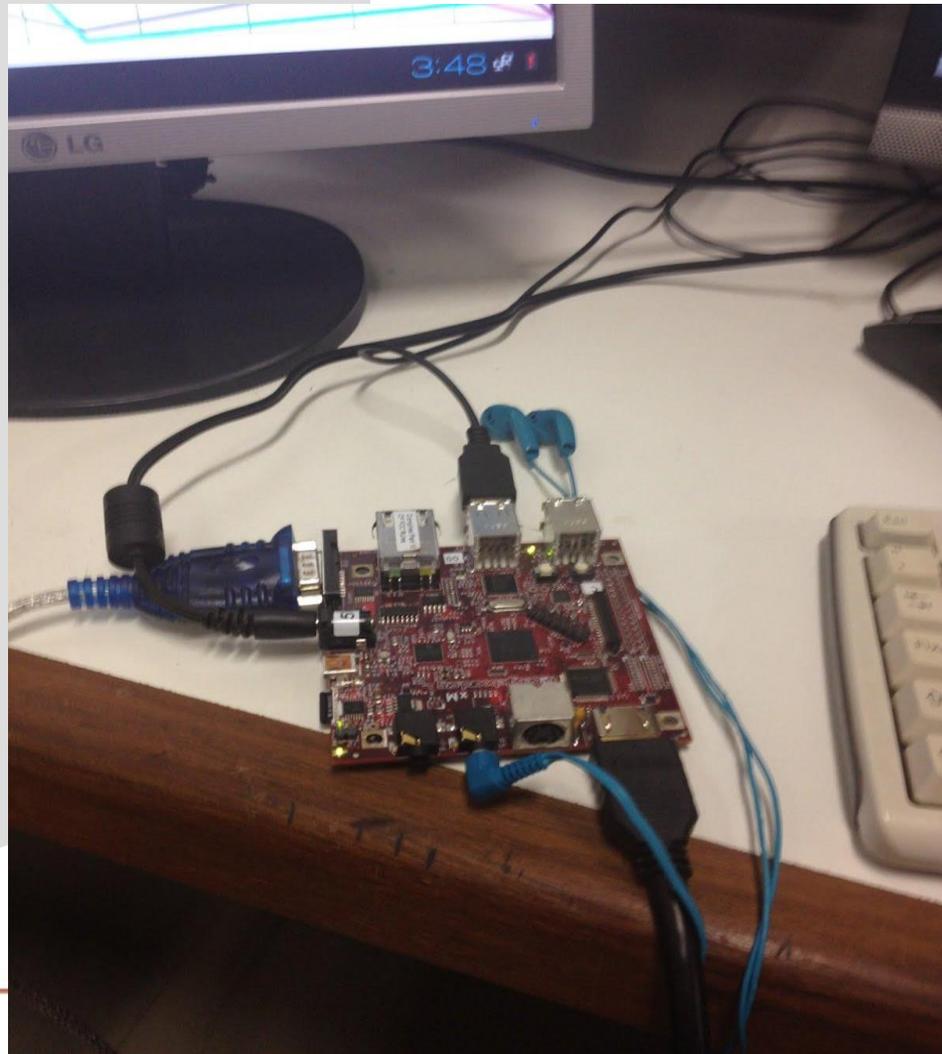
```
sleep 0.5
```

```
done
```

# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

	Música	Formato		
		MP3	Ogg	Wav
Instrumental	From a Distance	0,49	1,59	0,25888
	Juwita Malam	0,38765	0,4830	0,103641
	<b>MÉDIA:</b>	<b>0,44</b>	<b>1,04</b>	<b>0,18</b>
				<b>0,55</b>
RAP	Love is Blue	0,48262	1,37986	0,02288
	Don't Lie	0,445455	0,8290	1,48577
	Hey Mama	1,1246	0,32497	1,78652
	<b>MÉDIA:</b>	<b>0,68</b>	<b>0,84</b>	<b>1,10</b>
			<b>0,88</b>	
R&B	Boys and Men	0,3386	0,54298	0,45070
	I will always love you	0,51008	0,03980	0,25768
	What can I do	0,000963	0,41811	0,00554
	<b>MÉDIA:</b>	<b>0,28</b>	<b>0,33</b>	<b>0,24</b>
			<b>0,28</b>	
Rock	Cure for the itch	2,12376	2,35266	0,09017
	Hit the floor	1,23829	1,53650	0,75852
	<b>MÉDIA:</b>	<b>1,68</b>	<b>1,94</b>	<b>0,42</b>
				<b>1,35</b>

# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID



# 11. Análise de Consumo de Energia de Arquivos de Áudio no ARM com ANDROID

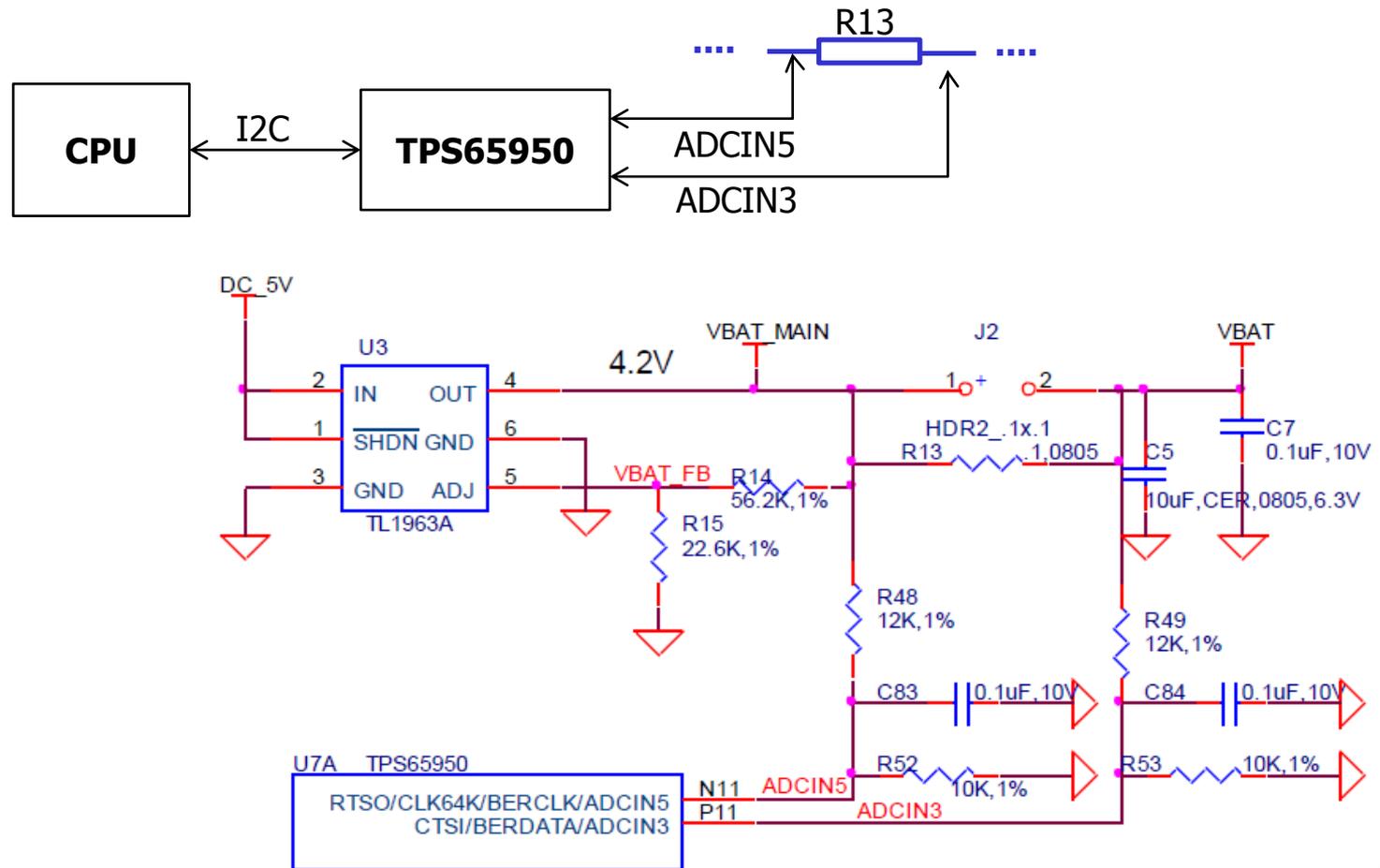
- Formato de áudio com **maior** média de consumo de potência: OGG
- Estilo musical com **maior** média de consumo de potência: ROCK
- Formato de áudio com **menor** média de consumo de potência: WAV
- Estilo musical com **menor** média de consumo de potência: R&B

# 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

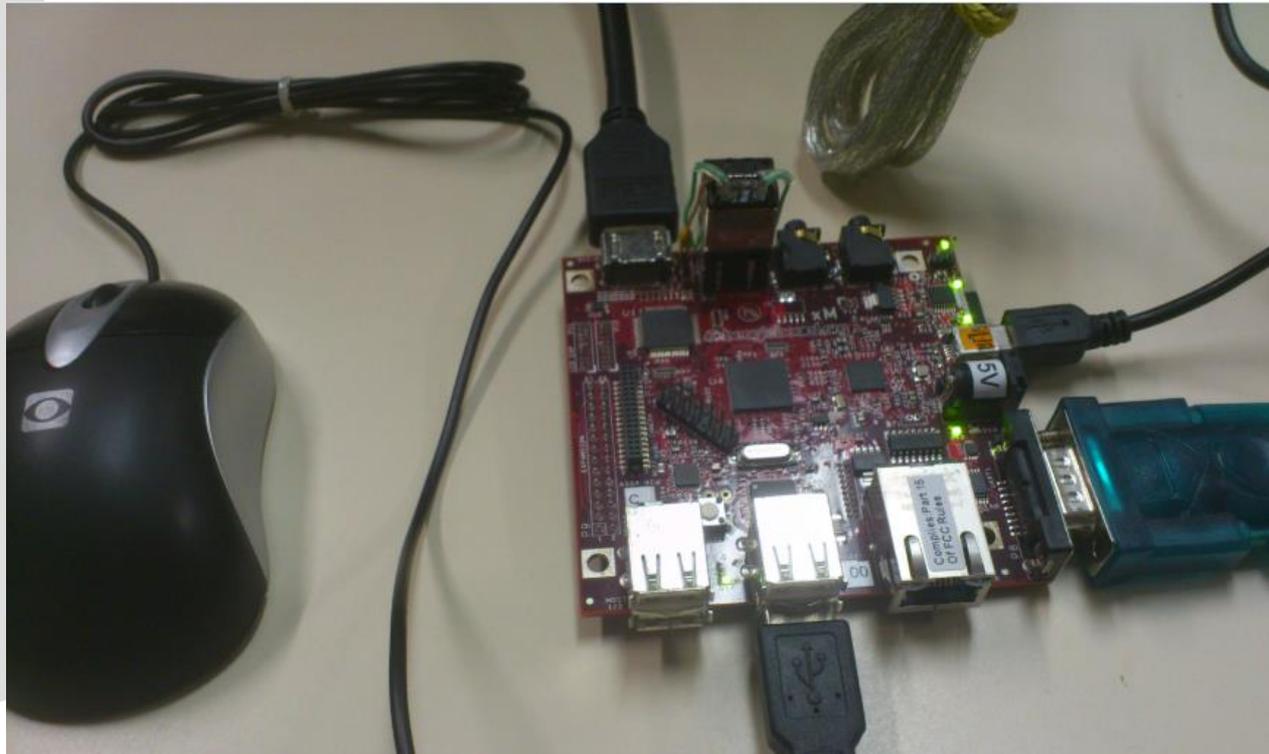
- Equipe
  - Antônio Barreto
  - Marcus Duarte
  - Paulo Guedes
  - Washington Azevedo
- Uso Android
  - Estimativa de Energia
- Uso da Beagleboard



# 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID



## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID



## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID



## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- Leitura de tensões via boot loader
  - U-boot possui comandos I2C
  - Determinação de:
    - Qual o bus I2C correto (há 3 buses)
    - Endereço de dispositivo no bus
    - Identificar os registradores necessários
    - Identificar os comandos e a ordem
  - Necessário ativar sinal de clock para leitura
  - Documentação não é clara neste ponto

## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- Leitura de tensões via boot loader: configuração
  - //Gravar o GPBR1 ativando os bits MADC\_HFCLK\_EN(bit 7)
  - `i2c mw 0x49 0x91 0x90 1`
  - //Seta pra 1 o bit MADCON do reg. CTRL1
  - `i2c mw 0x4a 0x00 0x01 1`
  - // Seta pra 1 o bit SEL\_MAC\_MCPC do reg. CARKIT\_ANA\_CONTROL
  - `i2c mw 0x48 0xbb 0x08 1`
  - // Seta pra 1 os bits CH3 e CH5 do reg. SW1SELECT\_LSB
  - `i2c mw 0x4a 0x06 0x28 1`
  - `i2c mw 0x4a 0x07 0x00 1`
  - // Seta pra 1 apenas os bits CH3 e CH5 do reg. SW1SELECT\_MSB
  - `i2c mw 0x4a 0x08 0x28 1`
  - `i2c mw 0x4a 0x09 0x00 1`

## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- Leitura de tensões via boot loader: Leitura dos sinais
  - //Seta pra 1 o bit SW1 do reg. CTRL\_SW1
  - `i2c mw 0x4a 0x12 0x20 1`
  - // Lê 2 bytes do reg GPCH3\_LSB + MSB
  - `i2c md 0x4a 0x3d 2 003d: c0 50 .P`
  - // Lê 2 bytes do reg GPCH5\_LSB + MSB
  - `i2c md 0x4a 0x41 2 0041: 80 50 .P`

## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

0xc2c0, 0xc480
0xc2c0, 0xc480
0xc300, 0xc4c0
0xc280, 0xc480
0xc2c0, 0xc4c0
0xc340, 0xc500
0xc2c0, 0xc400
0xc280, 0xc480
0xc340, 0xc480
0xc2c0, 0xc440
0xc200, 0xc340
0xc2c0, 0xc480
0xc280, 0xc3c0
0xc240, 0xc400
0xc280, 0xc3c0
0xc240, 0xc400
0xc340, 0xc4c0
0xc2c0, 0xc440

0xc300, 0xc480
0xc2c0, 0xc4c0
0xc280, 0xc440
0xc180, 0xc2c0
0xc300, 0xc400
0xc2c0, 0xc400
0xc300, 0xc480
0xc280, 0xc480
0xc240, 0xc440
0xc2c0, 0xc440
0xc2c0, 0xc480
0xc340, 0xc4c0
0xc2c0, 0xc400
0xc300, 0xc4c0
0xc280, 0xc400
0xc300, 0xc480
0xc2c0, 0xc400
0xc300, 0xc4c0

0xc2c0, 0xc480
0xc300, 0xc4c0
0xc300, 0xc4c0
0xc340, 0xc480
0xc300, 0xc4c0
0xc340, 0xc540
0xc240, 0xc440
0xc300, 0xc480
0xc280, 0xc3c0
0xc240, 0xc440
0xc280, 0xc400
0xc2c0, 0xc4c0
0xc2c0, 0xc480
0xc340, 0xc440
0xc2c0, 0xc400
0xc300, 0xc4c0
0xc300, 0xc4c0
0xc340, 0xc500

0xc2c0, 0xc480
0xc280, 0xc3c0
0xc2c0, 0xc480
0xc300, 0xc4c0
0xc300, 0xc4c0
0xc340, 0xc4c0
0xc2c0, 0xc440
0xbf0, 0xc1c0
0xc2c0, 0xc4c0
0xc300, 0xc480
0xc300, 0xc4c0
0xc280, 0xc400
0xc280, 0xc4c0
0xc340, 0xc4c0
0xc2c0, 0xc4c0
0xc300, 0xc4c0
0xc2c0, 0xc4c0
0xc2c0, 0xc440

## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- I2CTools
  - I2c-tools é um conjunto de programas que tornam fácil depurar dispositivos I<sup>2</sup>C sem ter que escrever qualquer código adicional.
  - Pode ser compilado no Linux
  - A BeagleBoard inclui suporte a I2C utilizando I2CTools

## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- Leitura de tensões no sistema operacional
  - Obter um kernel com suporte a I2C
  - Compilação do i2ctools
  - Conversão da sintaxe dos comandos
    - U-boot -> i2cget/i2cset
    - Identificadores do barramento mudam
      - 0, 1, 2 -> 1, 2, 3
  - Leitura de um único sinal: ok

Leitura de tensões (S.O.):  
configuração

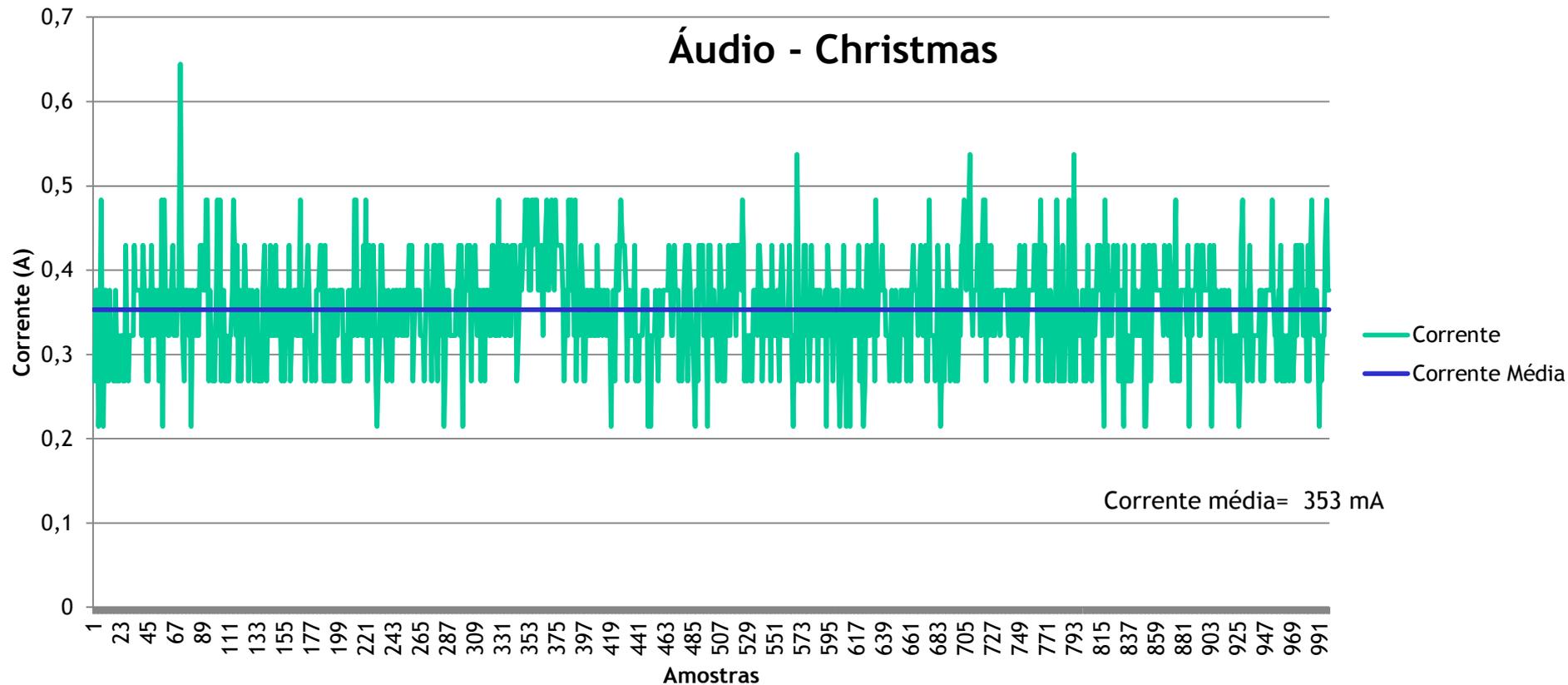
```
i2cset -f -y 1 0x49 0x91 0x90
i2cset -f -y 1 0x49 0x91 0x00
i2cset -f -y 1 0x49 0x91 0x90
i2cset -f -y 1 0x4a 0x00 0x01
i2cset -f -y 1 0x48 0xbb 0x08
i2cset -f -y 1 0x4a 0x06 0x28
i2cset -f -y 1 0x4a 0x07 0x00
i2cset -f -y 1 0x4a 0x08 0x28
i2cset -f -y 1 0x4a 0x09 0x00
```

## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- Leitura de tensões (S.O.): amostragem
  - `i2cset -f -y 1 0x4a 0x12 0x20`
  - `i2cget -f -y 1 0x4a 0x3d`
  - `i2cget -f -y 1 0x4a 0x41`
- Dispara a conversão AD
  - Lê o primeiro byte
  - Lê o segundo byte

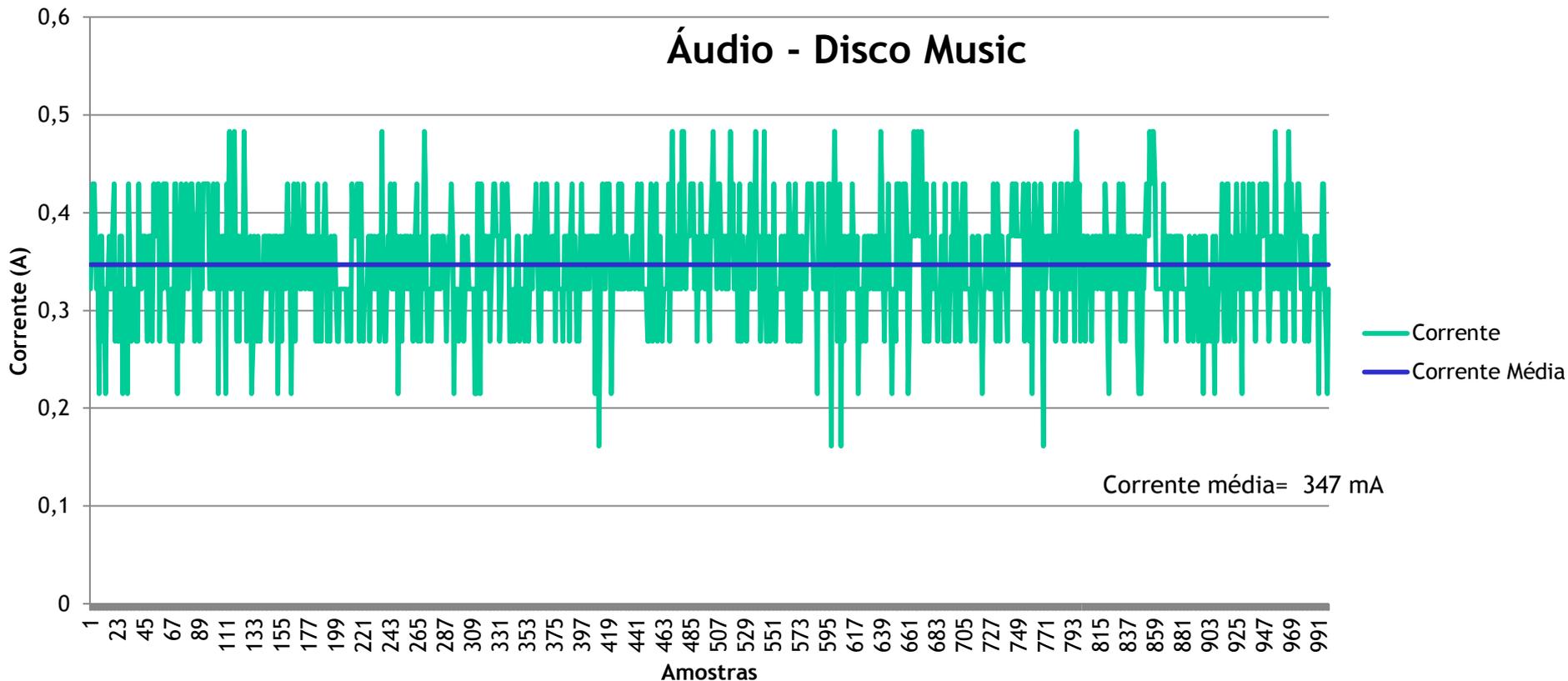
## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- Valores de Corrente (A) para aplicativos de Áudio



# 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

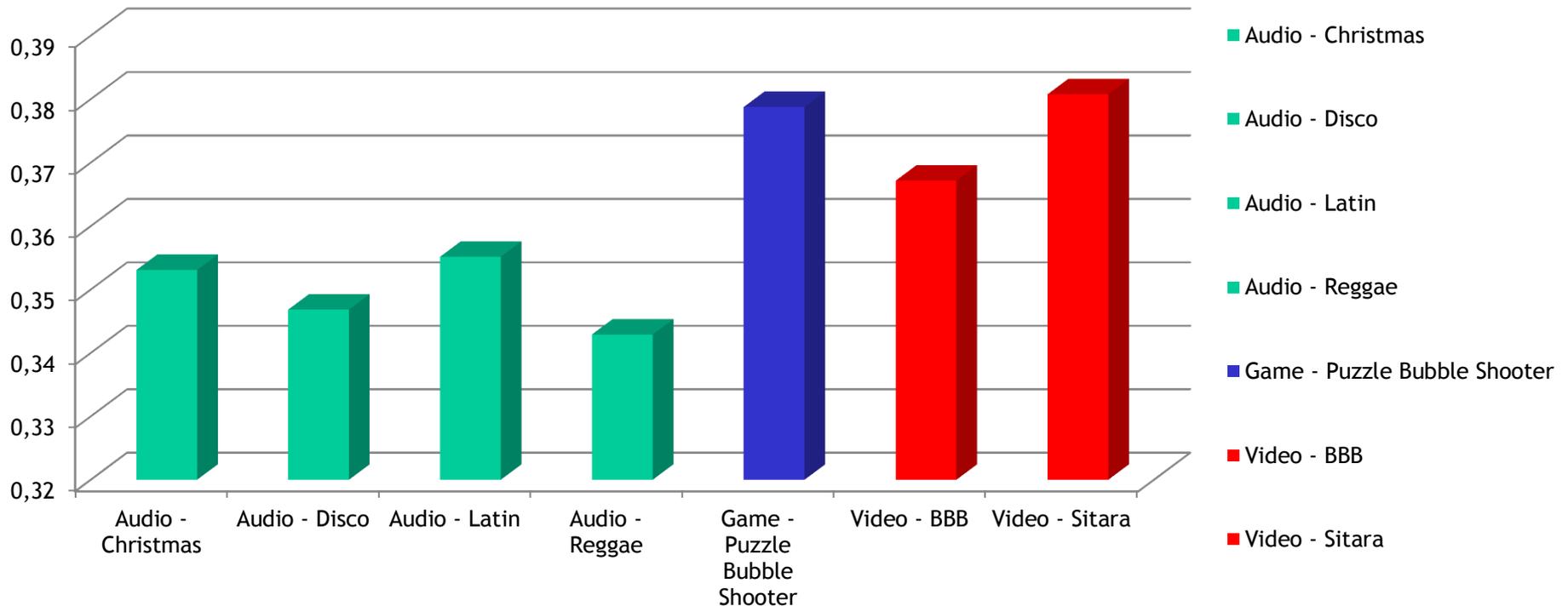
- Valores de Corrente (A) para aplicativos de Áudio



# 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

- Histograma das correntes médias

Corrente (A)

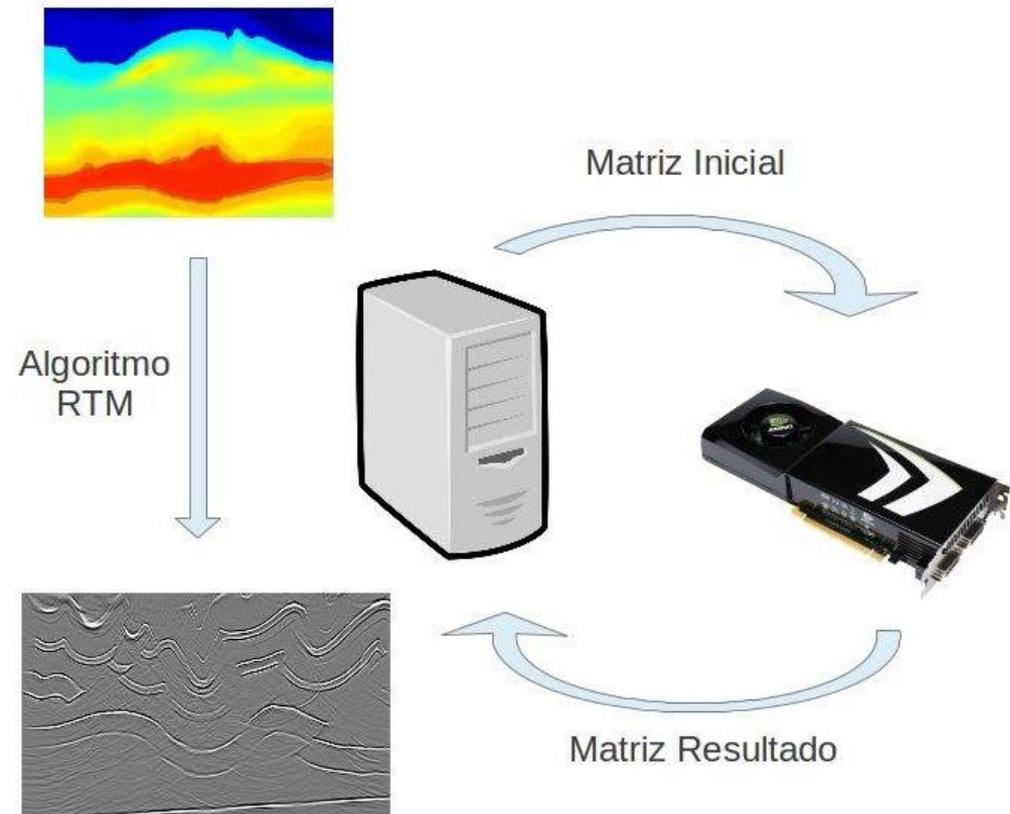


## 12. Estudo de Consumo de Energia de Aplicativos usando Plataforma BEAGLEBOARD-ANDROID

1. É possível medir o consumo de energia via software na própria plataforma;
2. Estratégia de medição pode ter um grande impacto na carga da CPU;
3. Observamos que usar a placa com alimentação via USB, teremos instabilidades;
4. Dependendo da versão do android, isso poderá impactar no desempenho e medição de consumo de energia;

# 13. Avaliação de Consumo de Energia do Algoritmo Reverse Time Migration em GPU

- Equipe
  - Dário Soares
  - Lucas Torquato
- Uso GPU
  - Estimativa de Energia
- Avaliação de Energia



# 13. Avaliação de Consumo de Energia do Algoritmo Reverse Time Migration em GPU

- Implementação do RTM com CUDA:

- GPU utilizada:

- NVIDIA GeForce GTX 580;
- 128 cores.

- CPU utilizada:

- Intel Core i7.

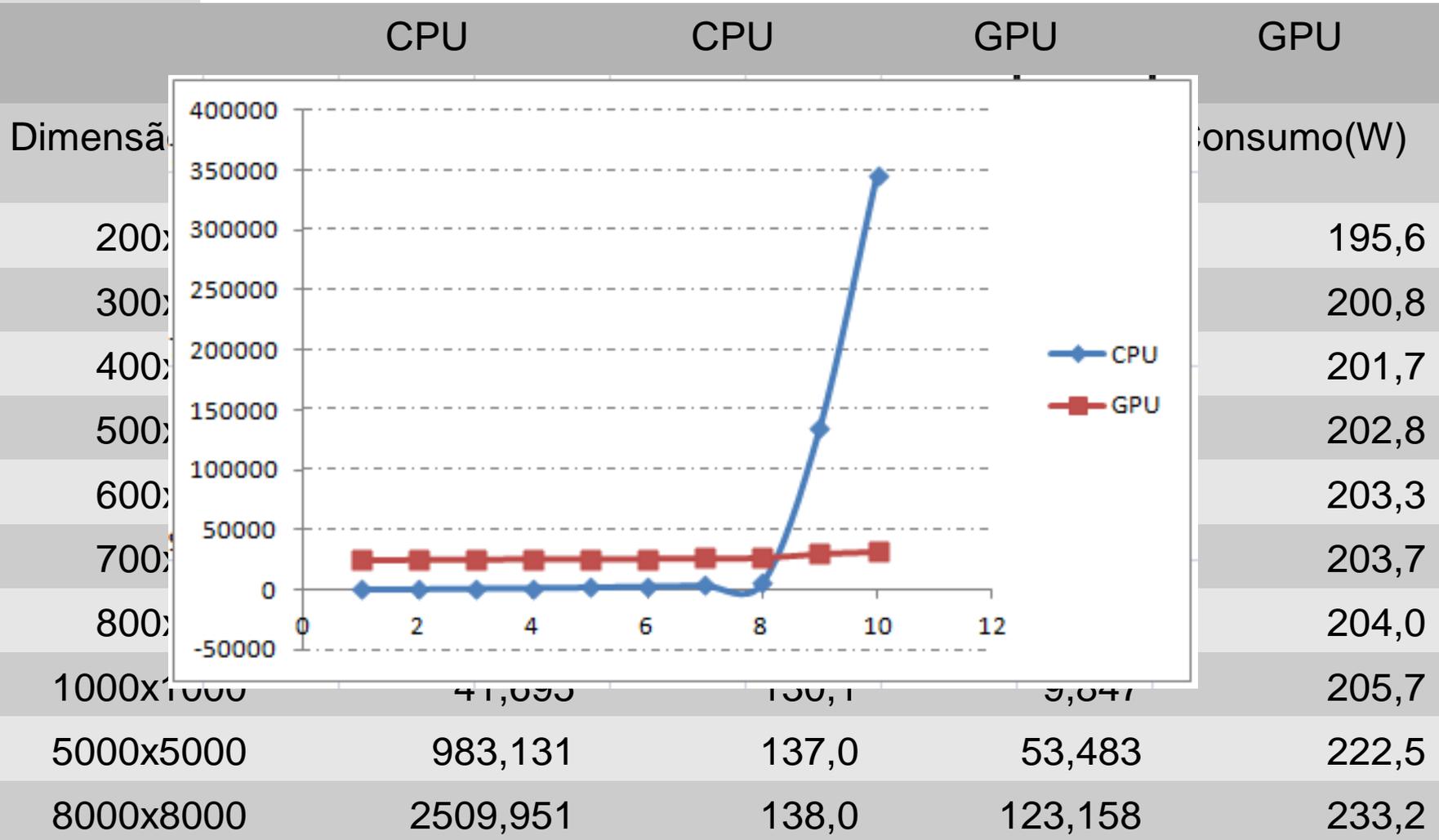
- Medição de consumo de energia:

- Feita com o WatsUp?.

- Os testes foram feitos com diferentes tamanhos de matrizes para que fosse possível analisar a relação speedup/consumo.

# 13. Avaliação de Consumo de Energia do Algoritmo Reverse Time Migration em GPU

- Comparação entre CPU e GPU:



## 13. Avaliação de Consumo de Energia do Algoritmo Reverse Time Migration em GPU

- Comparação entre CPU e GPU:
  - Em todos os casos, o tempo de execução foi medido desde o início da execução da aplicação;
  - O maior speedup atingido foi a execução 20 vezes mais rápida de GPU em relação à CPU, calculando uma imagem de 8000x8000 pixels.
  - Nesse caso, o consumo da GPU foi 1,6 vezes maior que o da CPU.