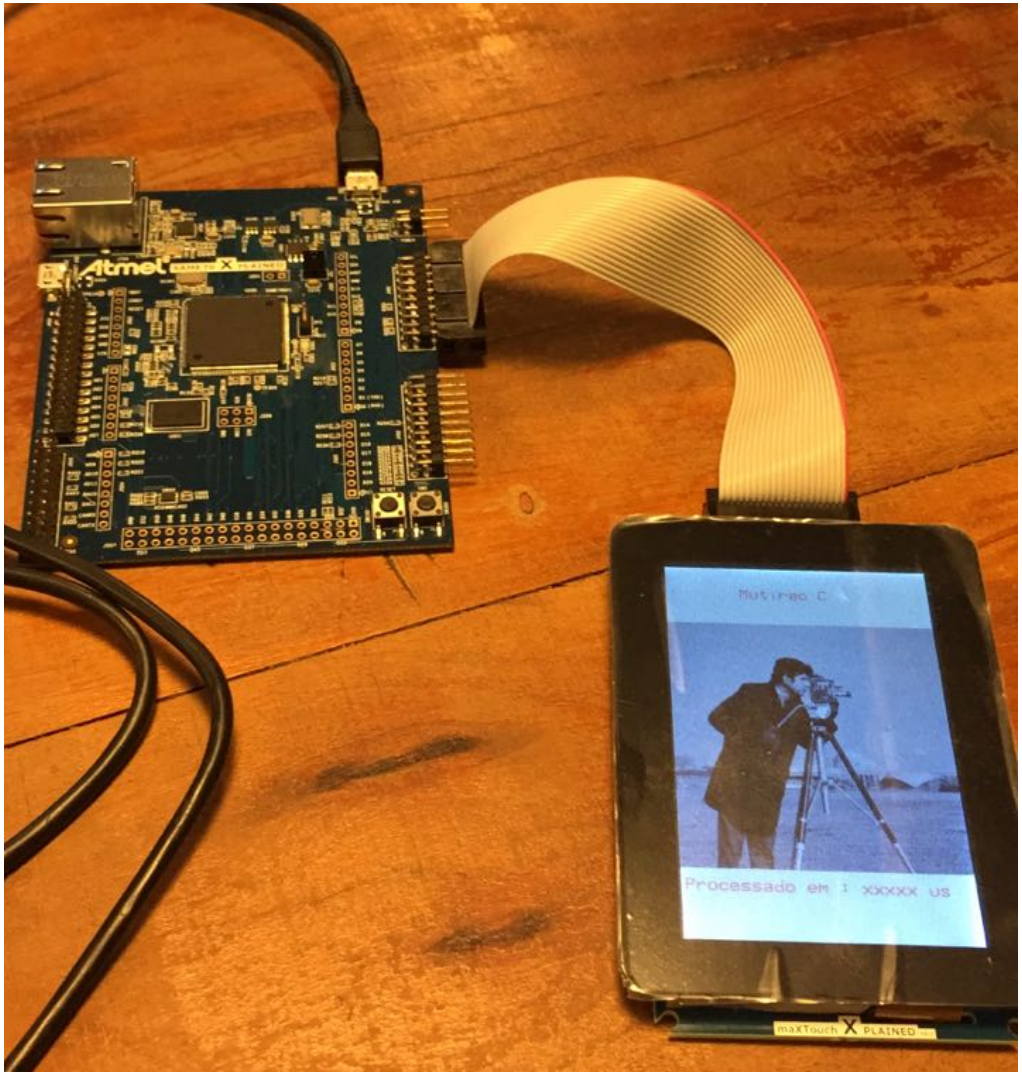
Mutirao C - 5s - Eng. da Computação - Aula 1

Rafael Corsi

2020

Mutirão C



Primeiros passos

Clonando

Clonem o repositório :

https://github.com/Insper/Labs-De-C

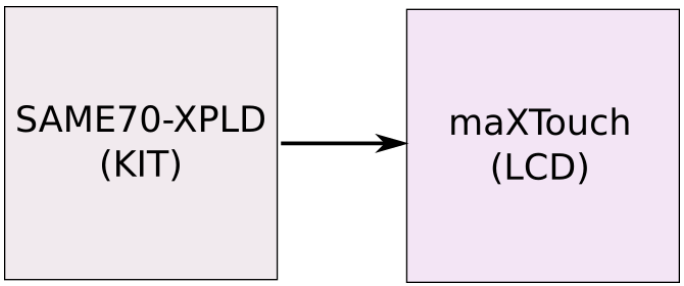
Sigam o roteiro para rodar o demo.

Parte 1: Seguir passos e parar!

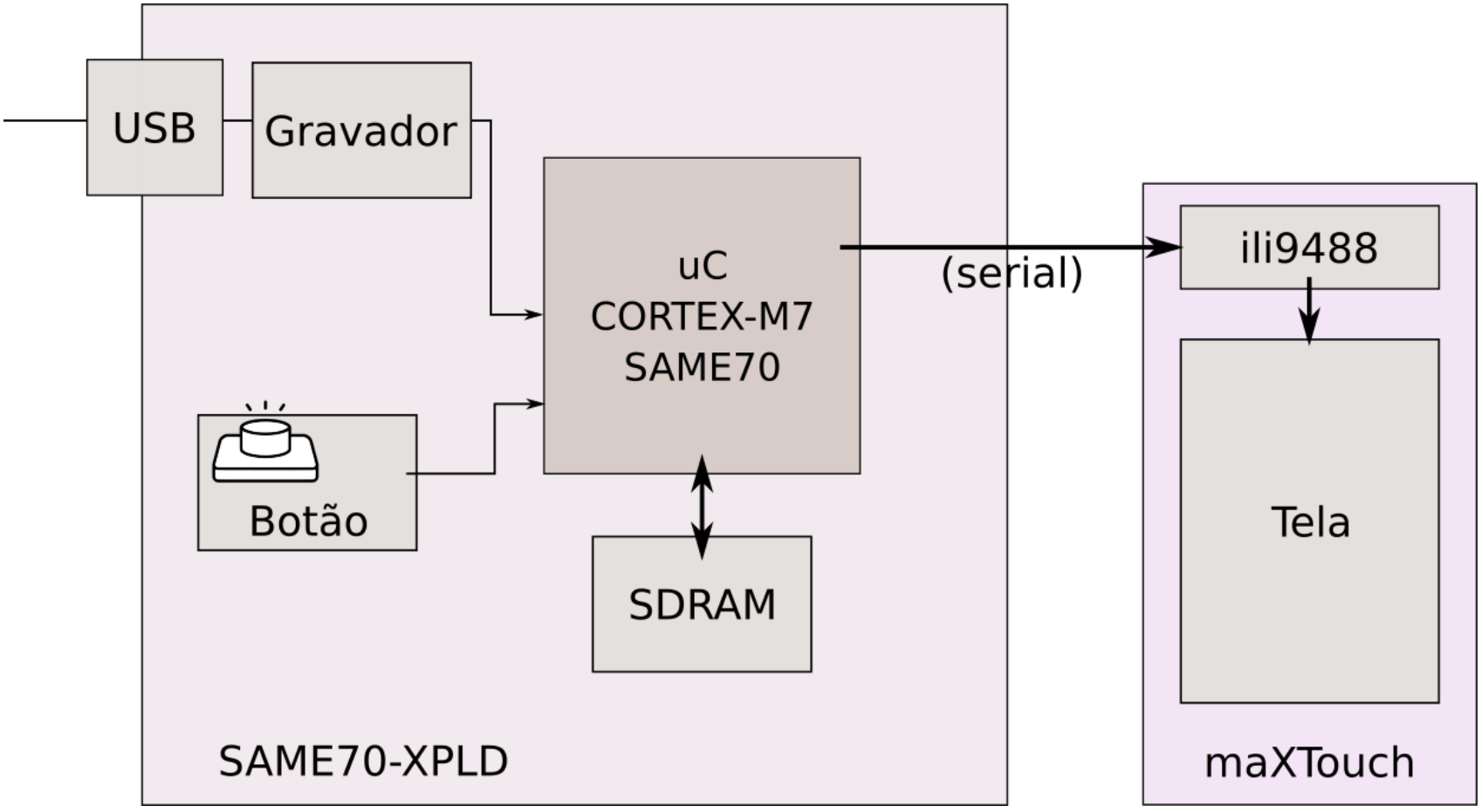
Parte 2: Entendendo

Visão geral

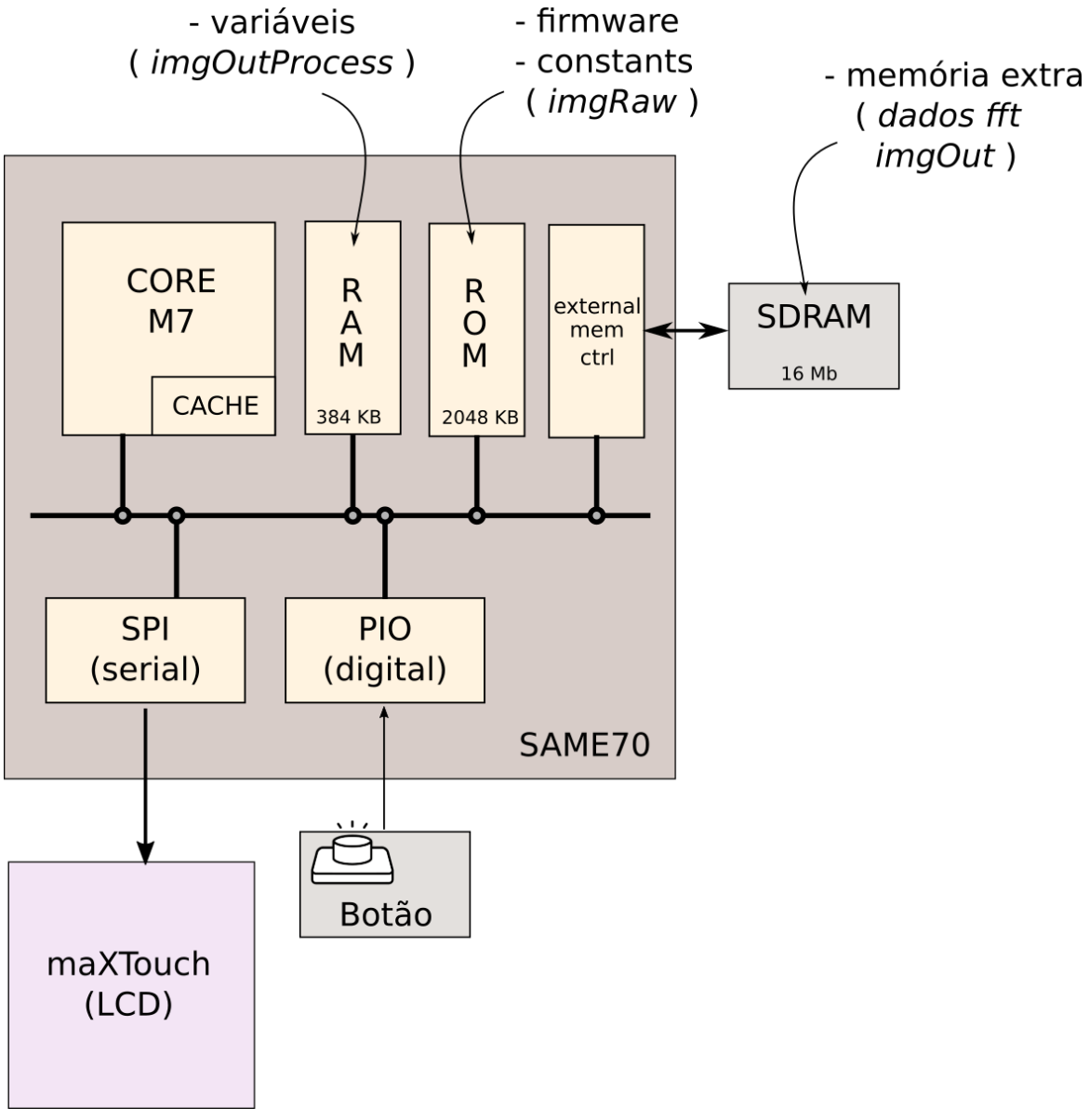
Em uma visão mais geral podemos analisar o sistema como um kit de desenvolvimento e um display LCD :



Hardware detalhado

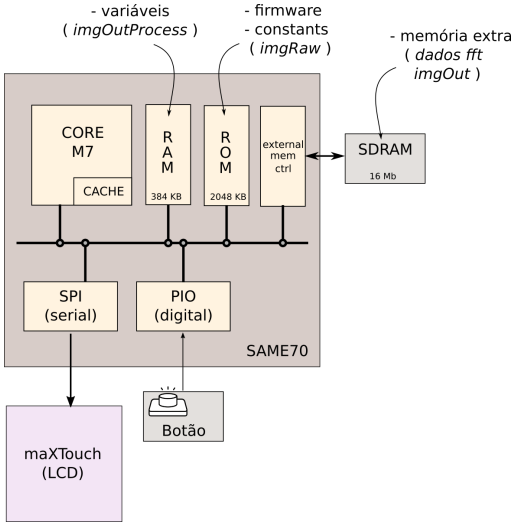


Microcontrolador

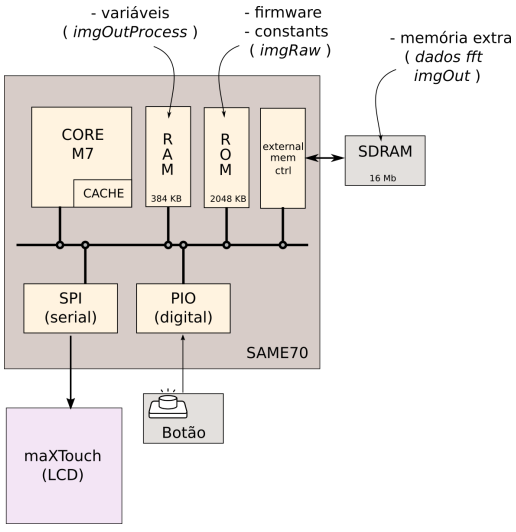


Muito lento não?

O que pode ser feito para melhorar?



O que pode ser feito para melhorar?



- **Clock**
- Alocar os dados em uma memória mais rápida
- DMA
- Mudar o protocolo de comunicação com o LCD
- ...

CLOCK

Frequência de operação atual: 4.5Mhz

src/config/conf_clock.h

```
// ===== Processor Clock (HCLK) Prescaler Options
//(Fhclk = Fsys / (SYSCLK_PRES))
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_32
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_3
```

- Fsys = 300Mhz

Alterando para 300 Mhz

src/config/conf_clock.h

```
// ===== Processor Clock (HCLK) Prescaler Options    (Fhclk
#define CONFIG_SYSCLK_PRES          SYSCLK_PRES_1
//#define CONFIG_SYSCLK_PRES        SYSCLK_PRES_2
//#define CONFIG_SYSCLK_PRES        SYSCLK_PRES_4
//#define CONFIG_SYSCLK_PRES        SYSCLK_PRES_8
//#define CONFIG_SYSCLK_PRES        SYSCLK_PRES_16
//#define CONFIG_SYSCLK_PRES        SYSCLK_PRES_32
//#define CONFIG_SYSCLK_PRES        SYSCLK_PRES_64
//#define CONFIG_SYSCLK_PRES        SYSCLK_PRES_3
```

- Fsys = 300Mhz

Programe e veja a diferença

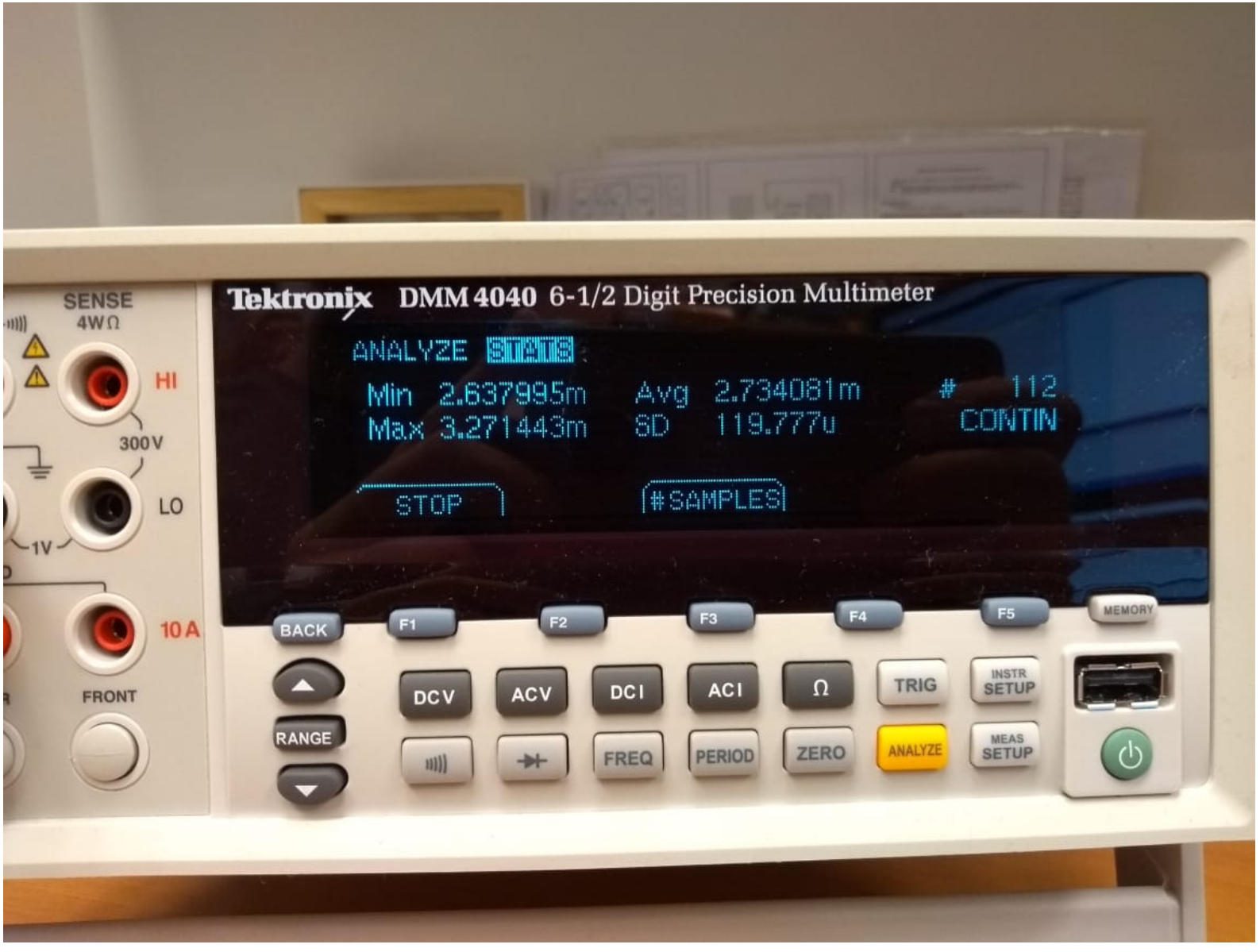
Impactos?

Energia em sistemas MOSFET

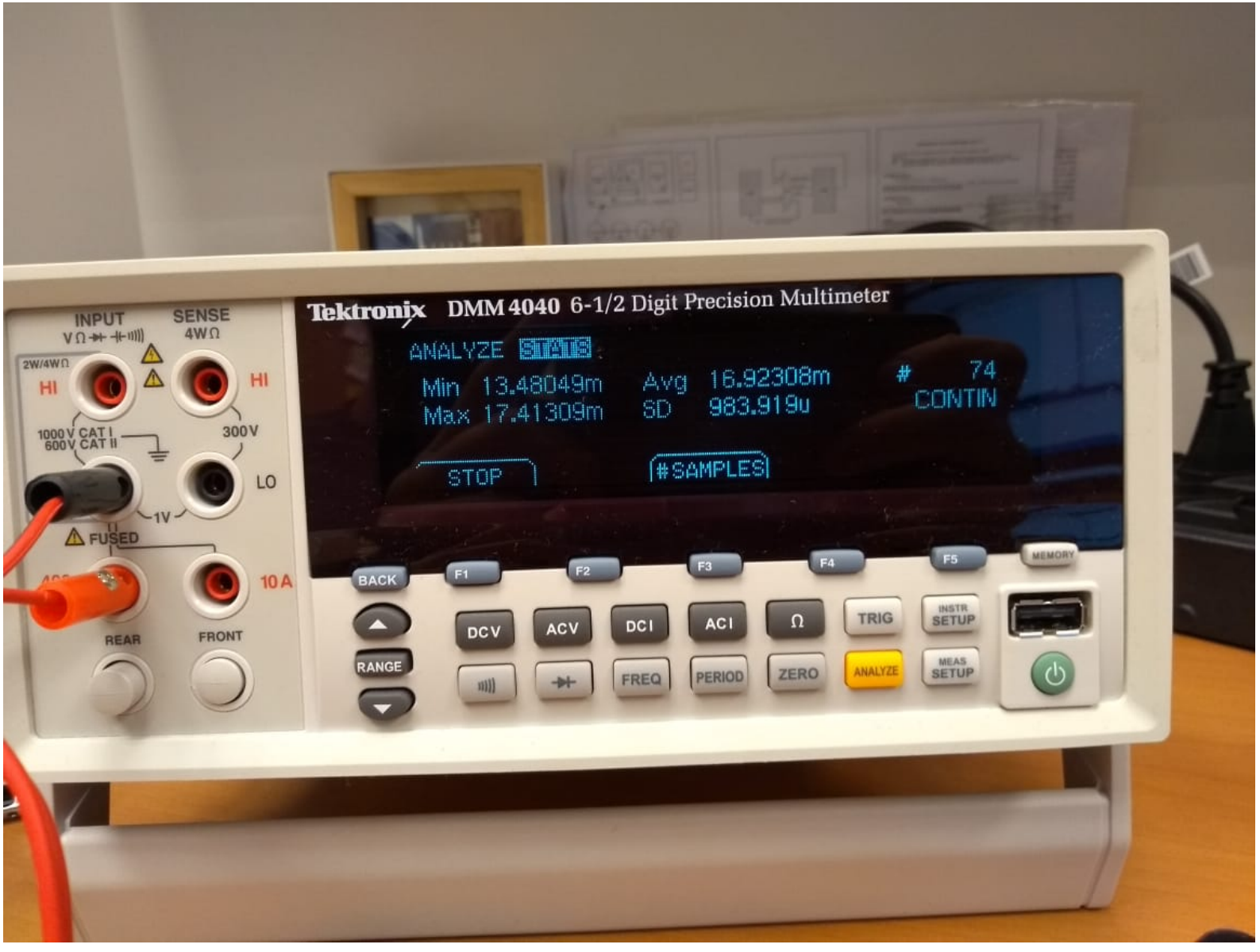
$$P = \alpha C V_{DD}^2 f$$

- alpha : fator de chaveamento (influenciado pelo código)
- C : capacitância
- V : tensão de operação
- f : frequência de chaveamento

Energia @ 4.6Mhz



Energia @ 300 Mhz



Aplicações

Similar?

Áreas em destaque

Keeping you safe at 70mph. Higher performance system platforms for connectivity, advanced recognition and decision intelligence will enable coming generations of assisted and automated driving.

Read more

Healthcare is becoming personal. Pervasive mobile computing, highly-efficient embedded intelligence and secure technology are driving new interaction capabilities and advancing diagnostic and treatment options.

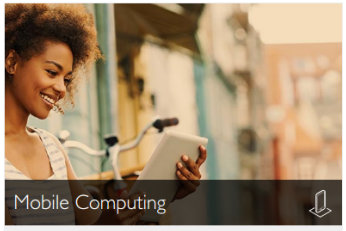
Read more

Leading compute density solutions are transforming infrastructure: Building an end-to-end enterprise application platform with increased intelligence for the delivery of fast, agile, service-based 5G networks, and delivering a paradigm shift in data center workload efficiency.

Read more

Enabling the foundations of IoT. Scalable embedded intelligence and software provides an agile, secure and flexible end-to-end platform for sensing, controlling and delivering business insights.

Read more

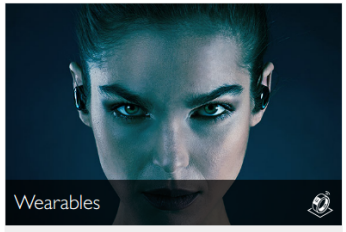


ARM low-power technology is the foundation of the mobile computing era, providing central processing capabilities as well as connectivity, and the industry's roadmap for evolving enhanced contextual, interactive and visual user experiences.

Read more

The combination of an integrated mobile experience and increasing intelligence and connectivity across devices in the home is creating opportunities to transform how we interact with our living spaces. More intuitive and interoperable systems save energy, increase security and safety, and reduce costs.

Read more

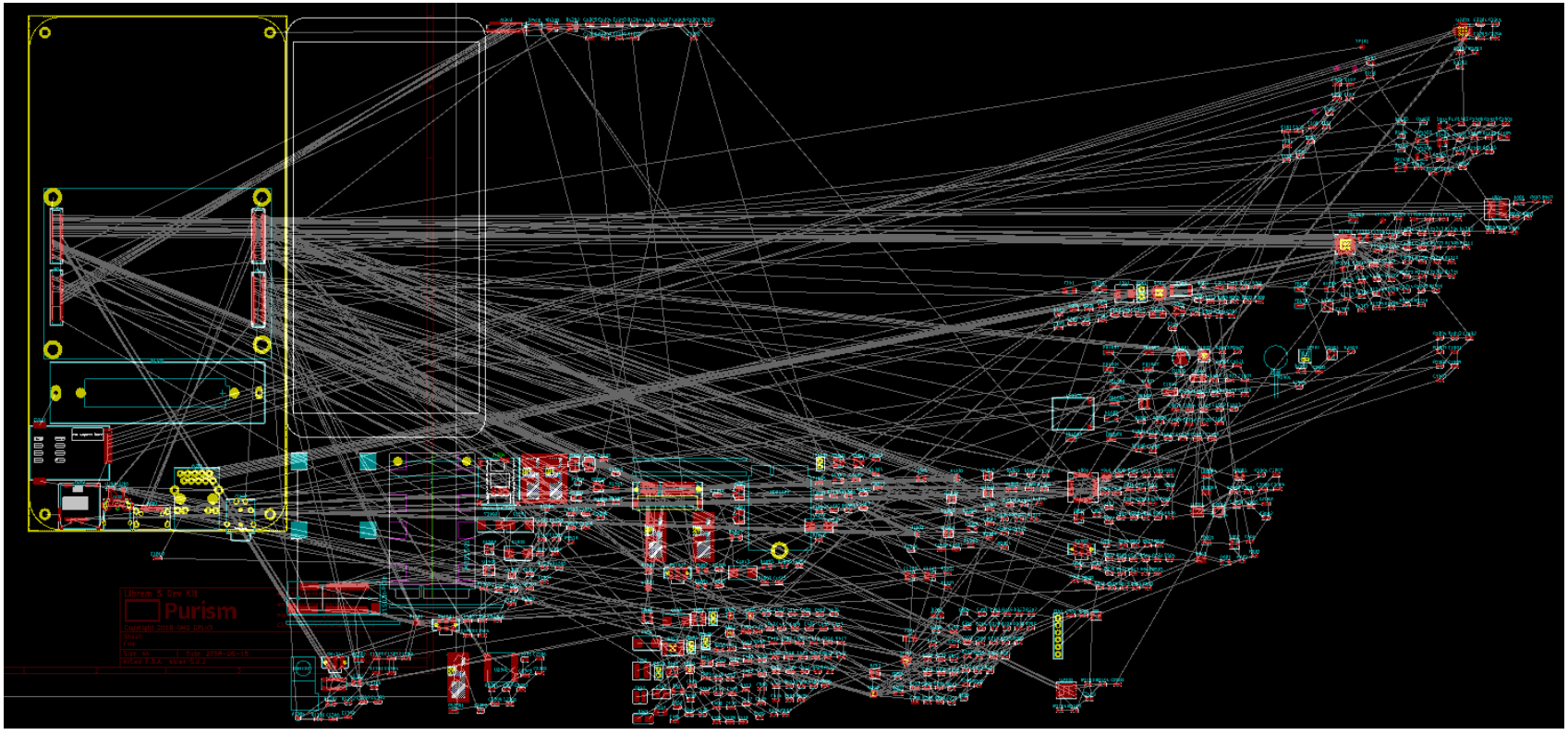


Redefining personal compute. Realizing new forms of interactions with our physical and digital worlds with highly-efficient processing for even the tiniest of devices.

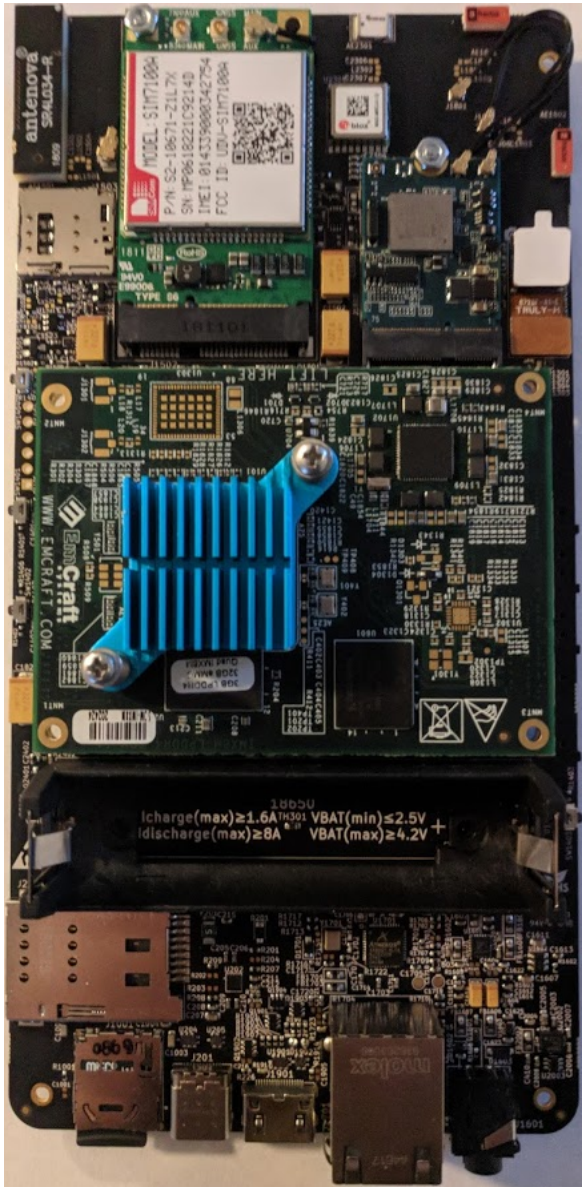
Celulares - Purism 5

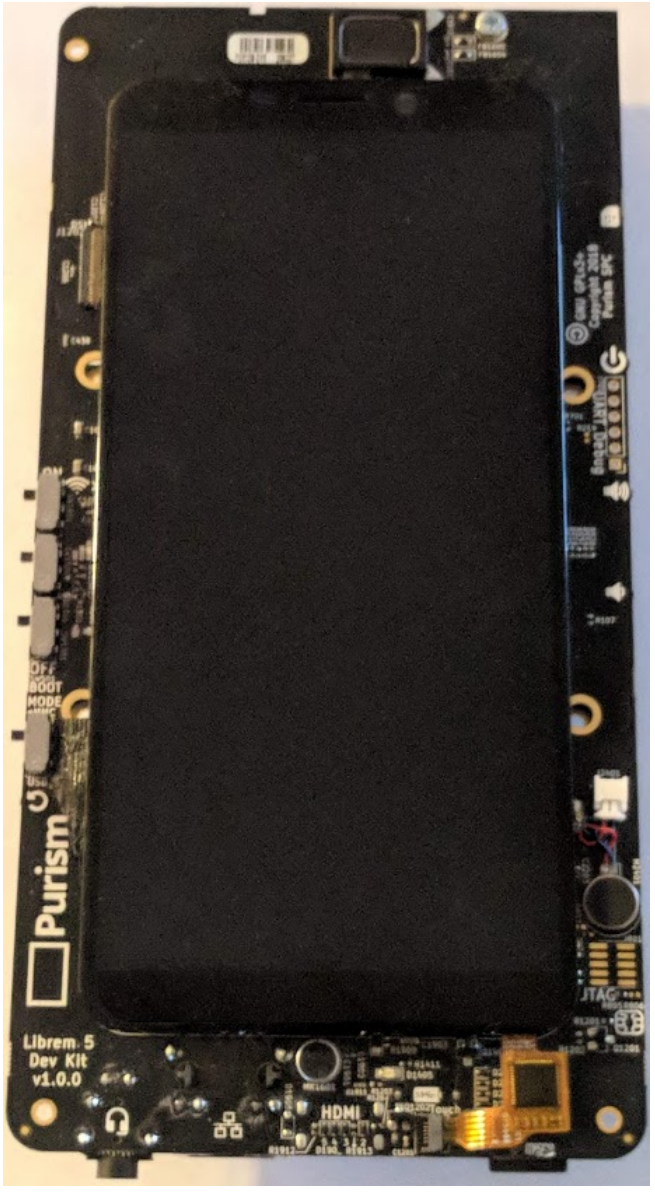
Protect your
private bits.

Celulares - Purism 5



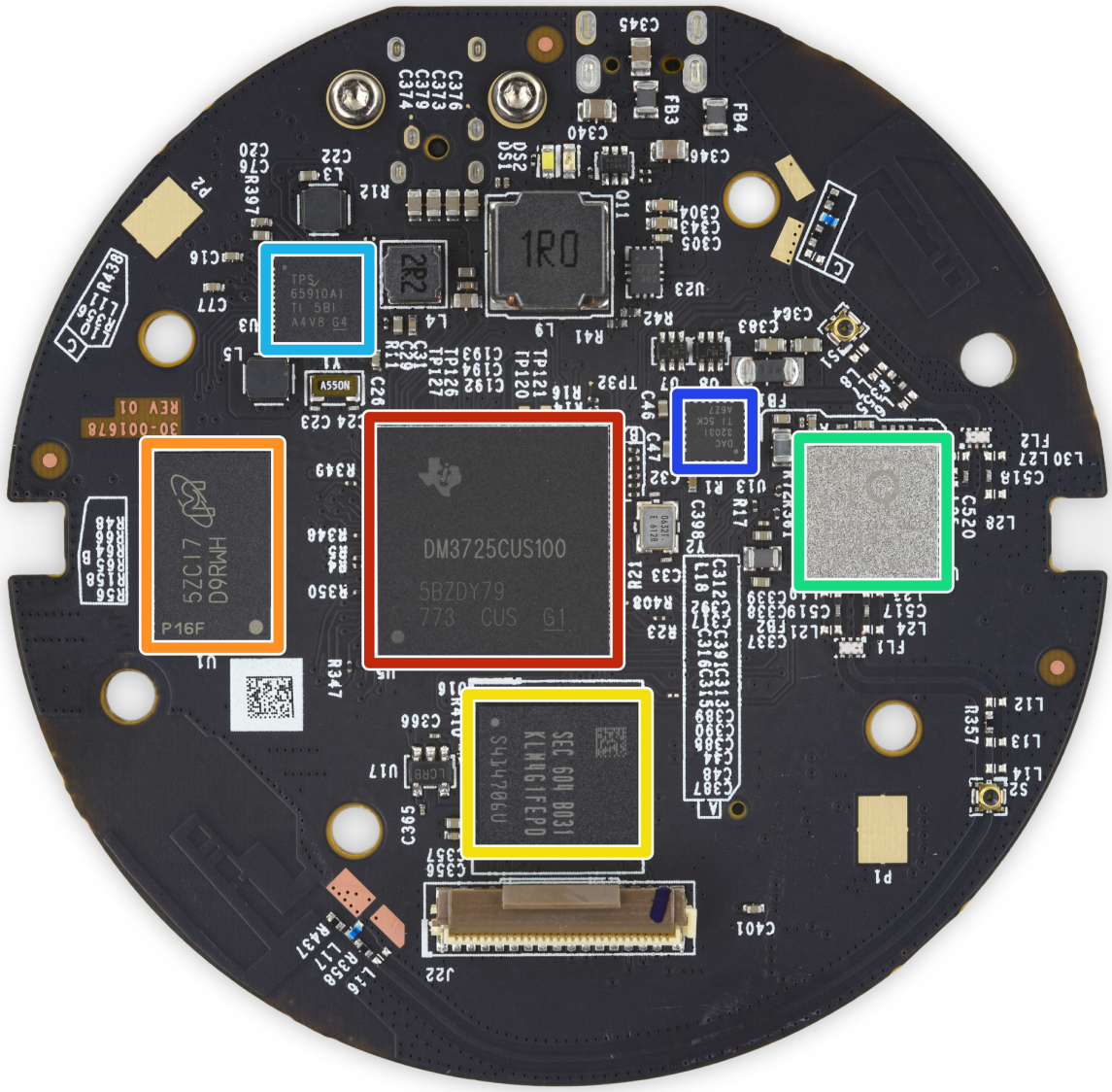
Celulares - Purism 5



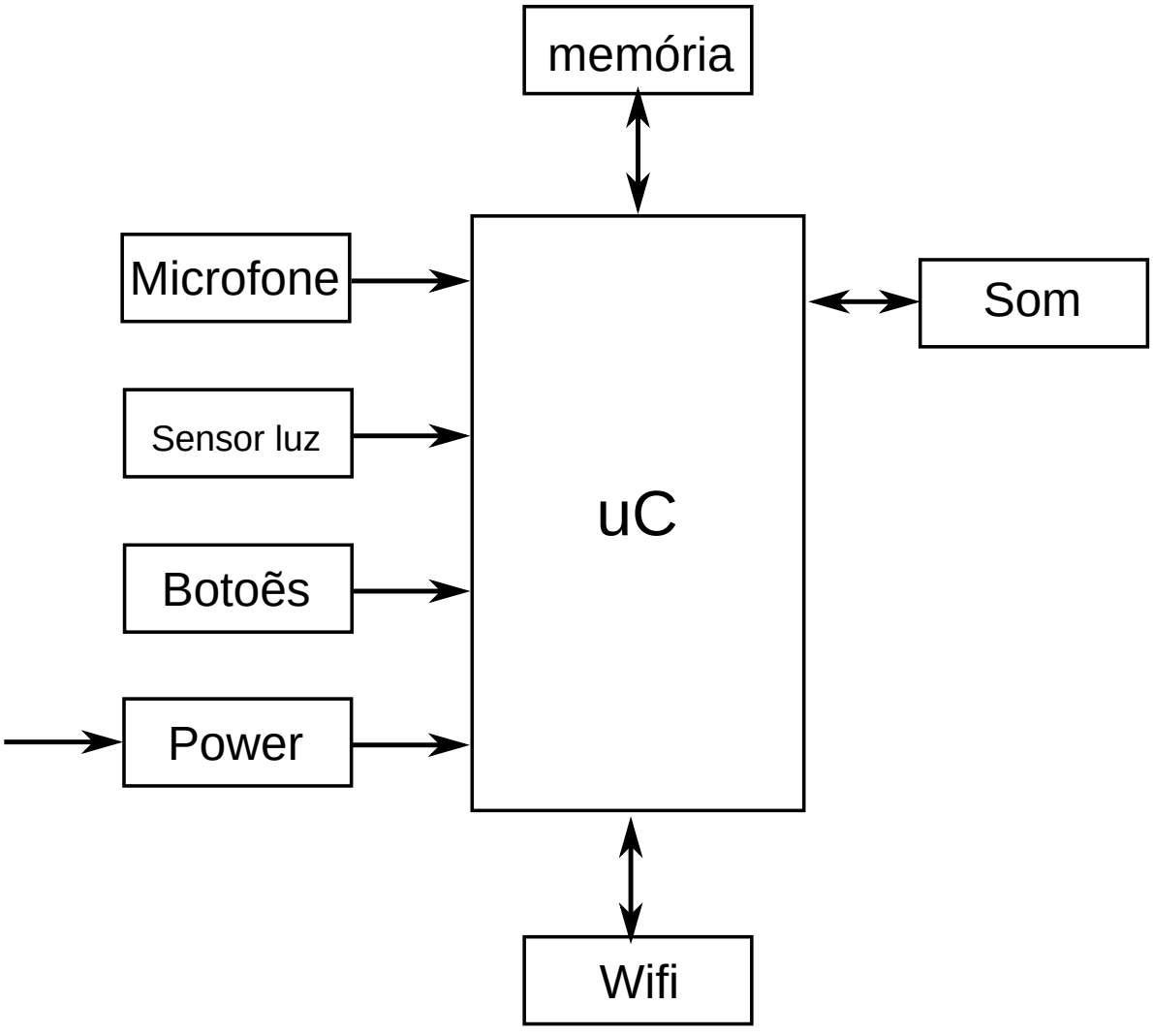


Amazon Echo Dot

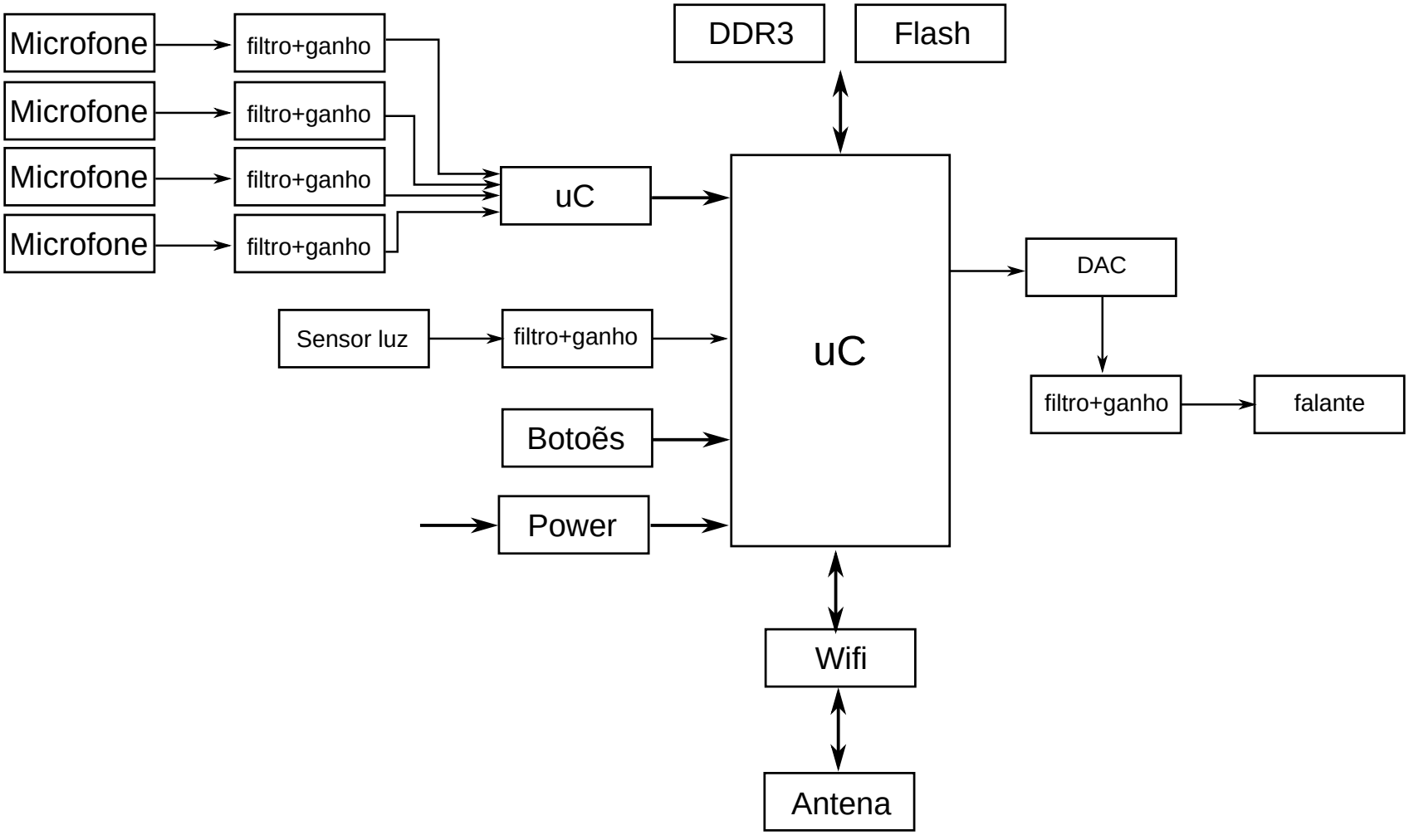
Echo Dot


TPS
65910A1
TI 5BI
A4V8 G4
1R0
2R2
A550N
DM3725CUS100
5BZDY79
773 CUS G1
5ZC17
D9RWH
P16F
SEC 604 B031
KLM4G1FEPD
REV 01
30-001678

Echo - Diagrama

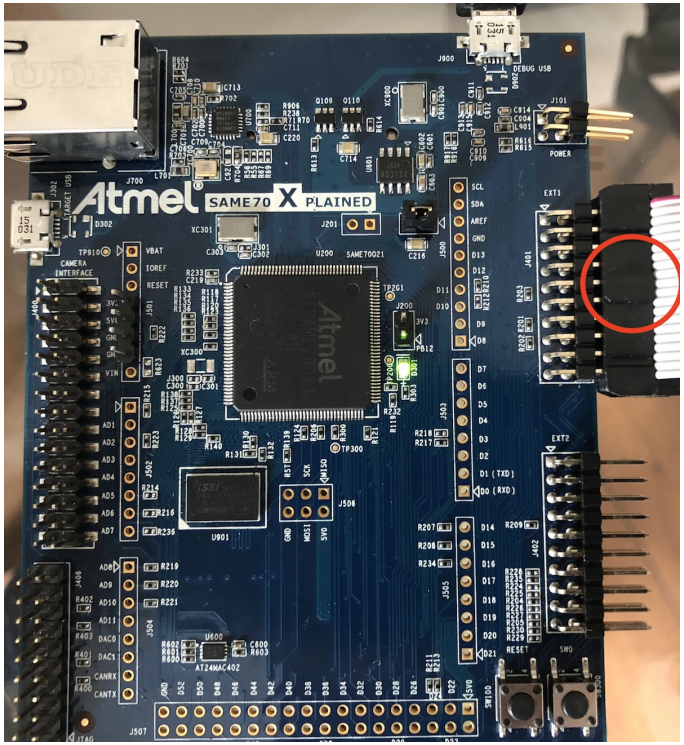


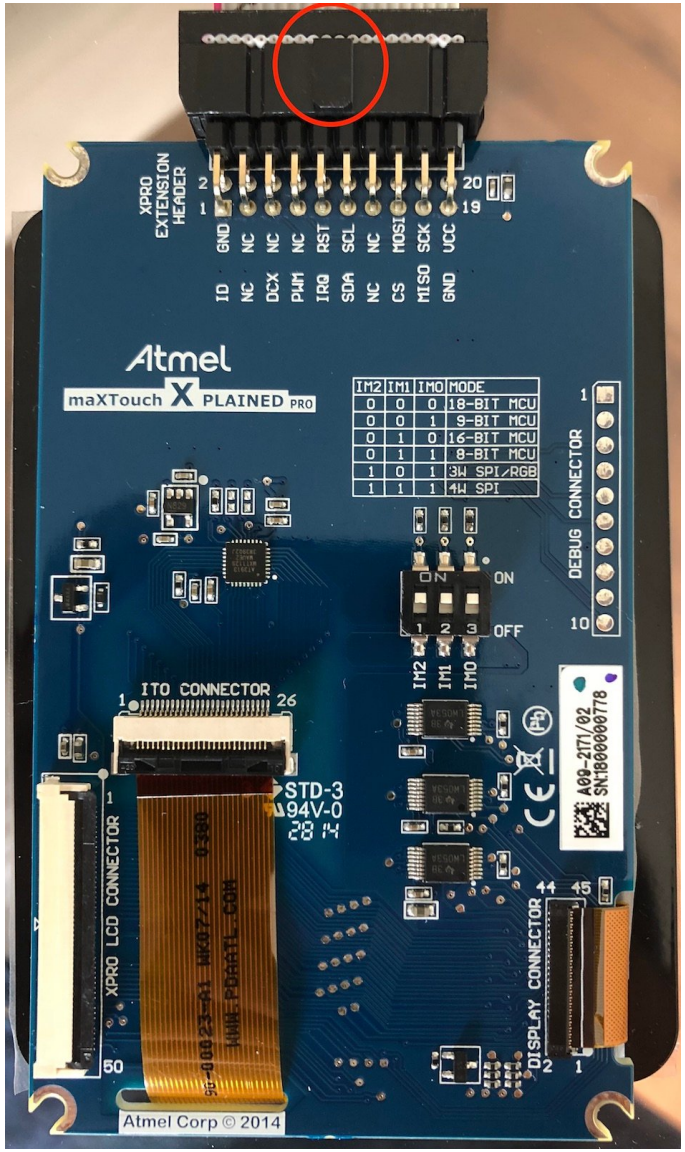
Echo - Diagrama detalhado



Extras

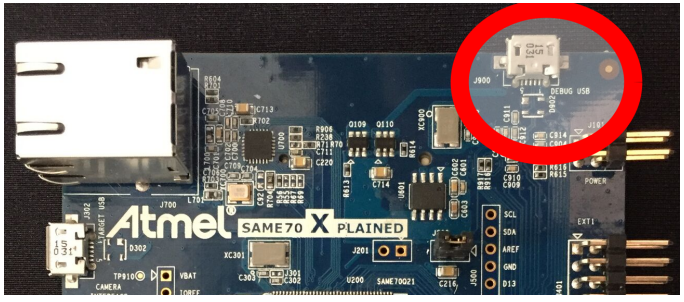
Primeiros passos, conectando.





Primeiros passos, rodando o exemplo.

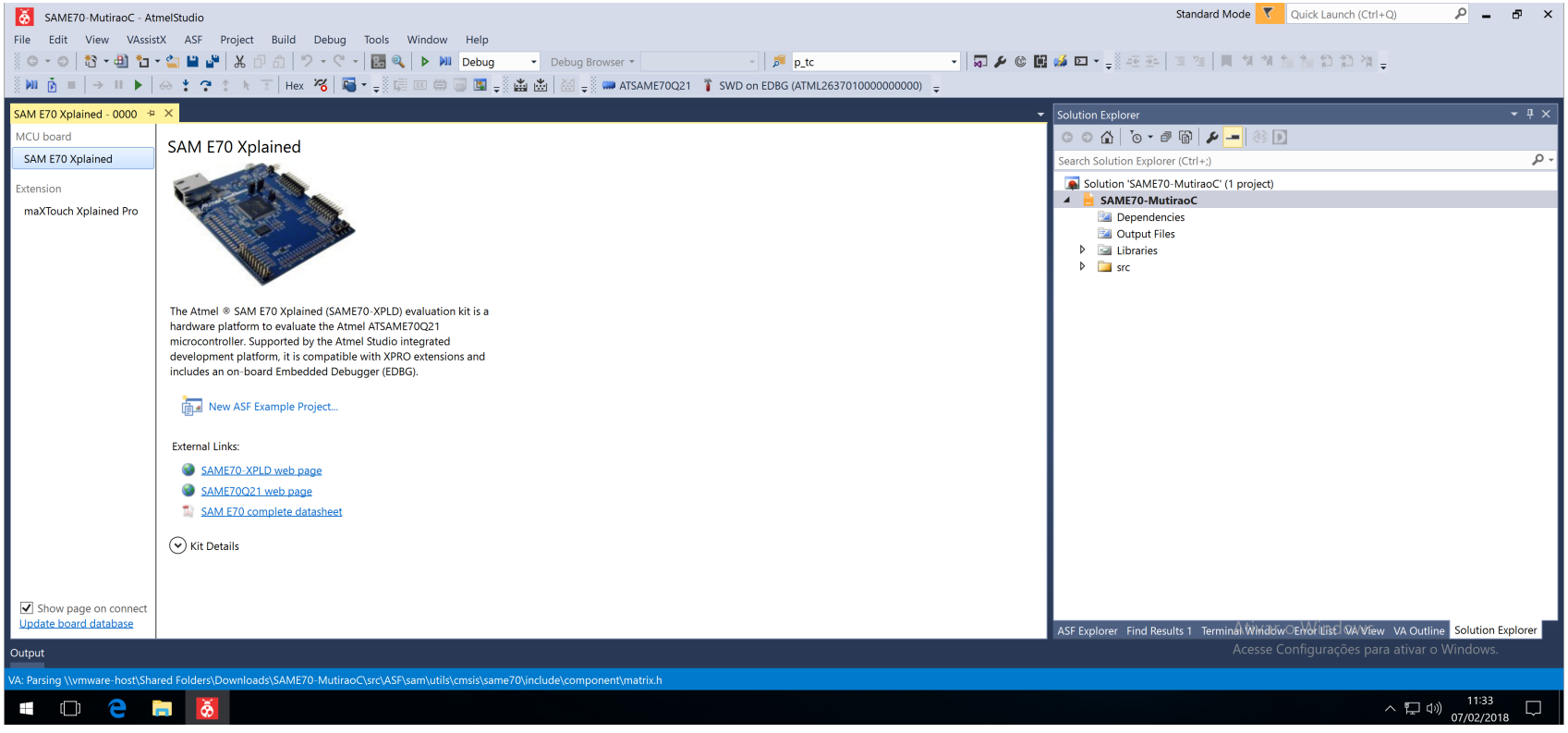
1. Conectar o USB do programador no computador



2. Abra o projeto exemplo localizado na pasta :

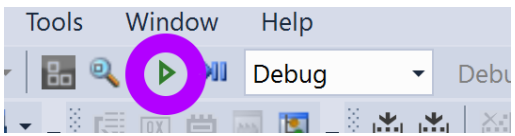
   01-DayOne/SAME70-MutiraoC/

Atmel Studio



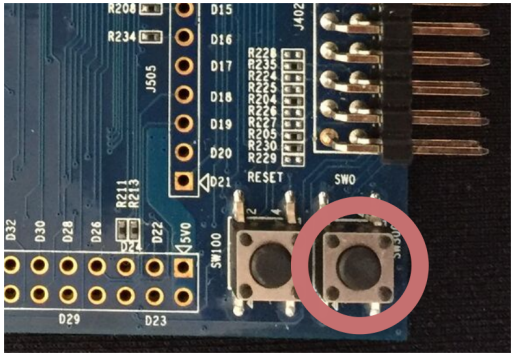
Programando

A etapa atual será a de embarcar o código exemplo no uC, para isso basta clicar em **Start Without Debug**



Botão

2. Uma vez embarcado o exemplo, o LCD deverá exibir uma imagem. A primeira imagem que aparece é a imagem original sem nenhum tipo de modificação, ao apertar o botão **SW0** do kit de desenvolvimento uma função **imageProcess()** é chamada e a imagem orignal é processada e então exibida.



Firmware

main

```
int main(){

    uint32_t time;
    uint8_t  imageSelect = 1;

    // inicializa placa e seus perifericos
    initBoardMutirao();

    // exibe imagem orignal, tempo de processamento suprim
    imgShow(imgRaw, 0);

    ...
```

SuperLoop

Aplicações embarcadas não devem nunca retornar do main.

```
while (1) {

  // se buttonFlag = 1 existe alteracao
  // no estado do botao
    if(buttonFlag){
```