Time Beta

Grupo Beta-4

Sistema de Controle do Canhão Fotônico

Felipe Zimmermann Homma
George Elias Ferreira da Silva
{felipe,rip}@inf.ufsc.br

http:// www . inf . ufsc . br /~ felipe /wiki

Objetivos

Time Beta

O **Time Beta** está encarregado do **Sistema de Defesa** , incluindo o **Radar** e o **Canhão Fotônico** .

Esse time vence o jogo destruindo a **Vulture Klingon** (e, infelizmente, matando **Kirk** e **Spock** no processo).

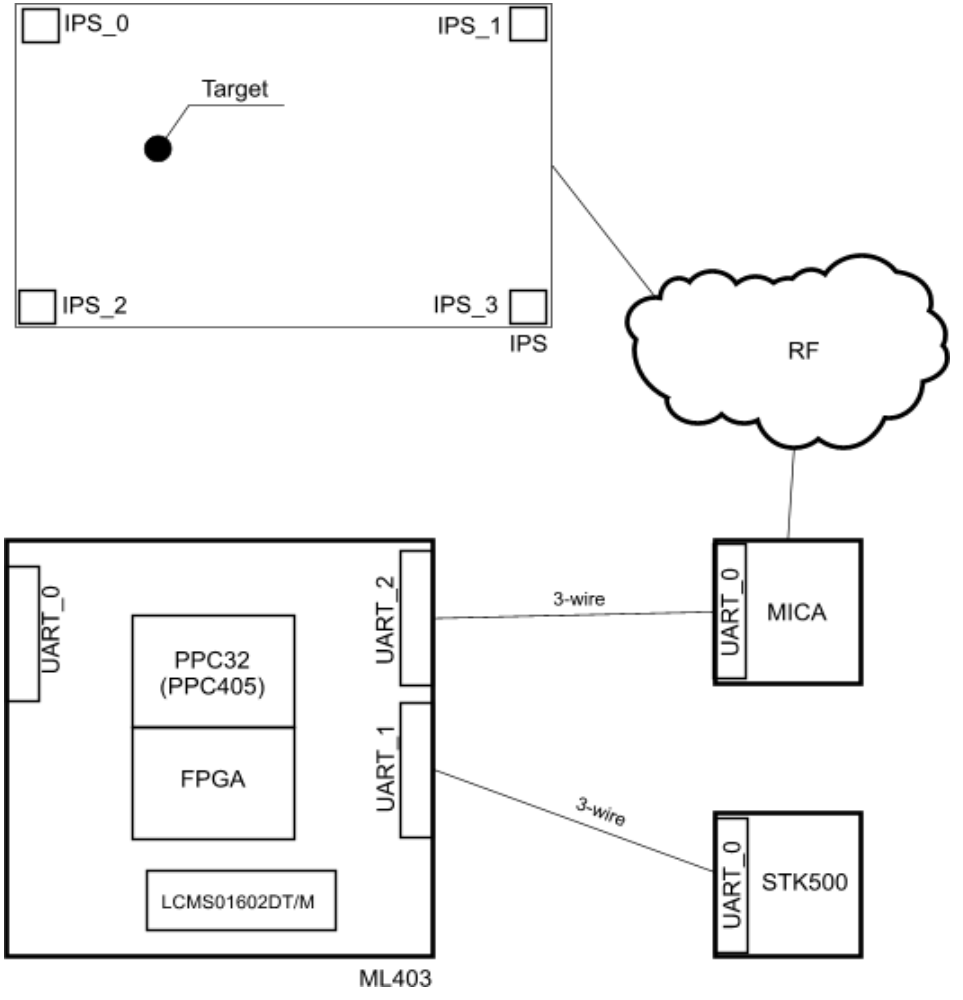
Grupo Beta 4

Este grupo devera desenvolver o **Programa de Controle** do **Sistema de Controle de Tiro** da **Enterprise** .

Nosso trabalho envolve fazer os trabalhos dos outros grupos (Beta 1 a 3) funcionarem em conjunto de forma a alcançar o objetivo primário (que é, infelizmente, matar **Kirk** e **Spock**).

Introdução

Visão Geral do Sistema



Essa visão esquemática apresenta o sistema que será contruído.

Embora 3 UARTs sejam sugeridas, apenas 2 são necessárias.

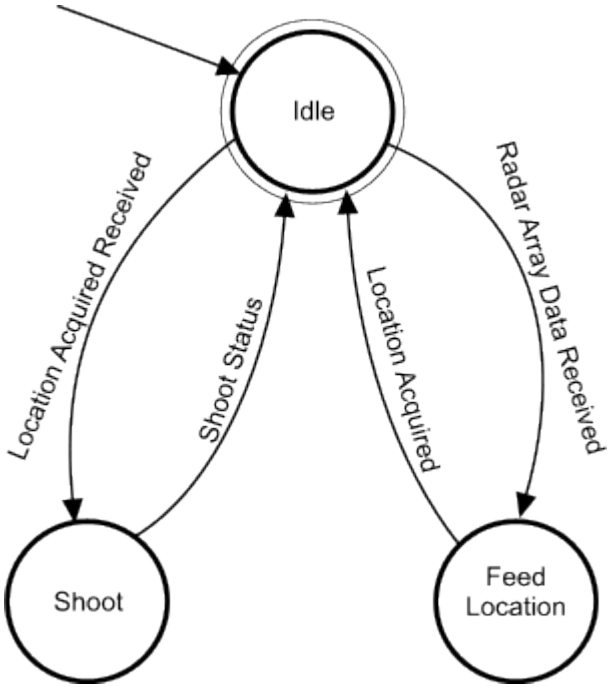
Introdução

Responsabilidades

- Capturar dados vindo do IPS
  - IPS->MICA (RF)
  - MICA->ML403 (UART)
- Disponibilizar facilidades do ML403
  - 16x2 Character Matrix LCD
  - UARTs (0,1,2)
- Integrar o restante do Time

Projeto

Primeira Tentativa



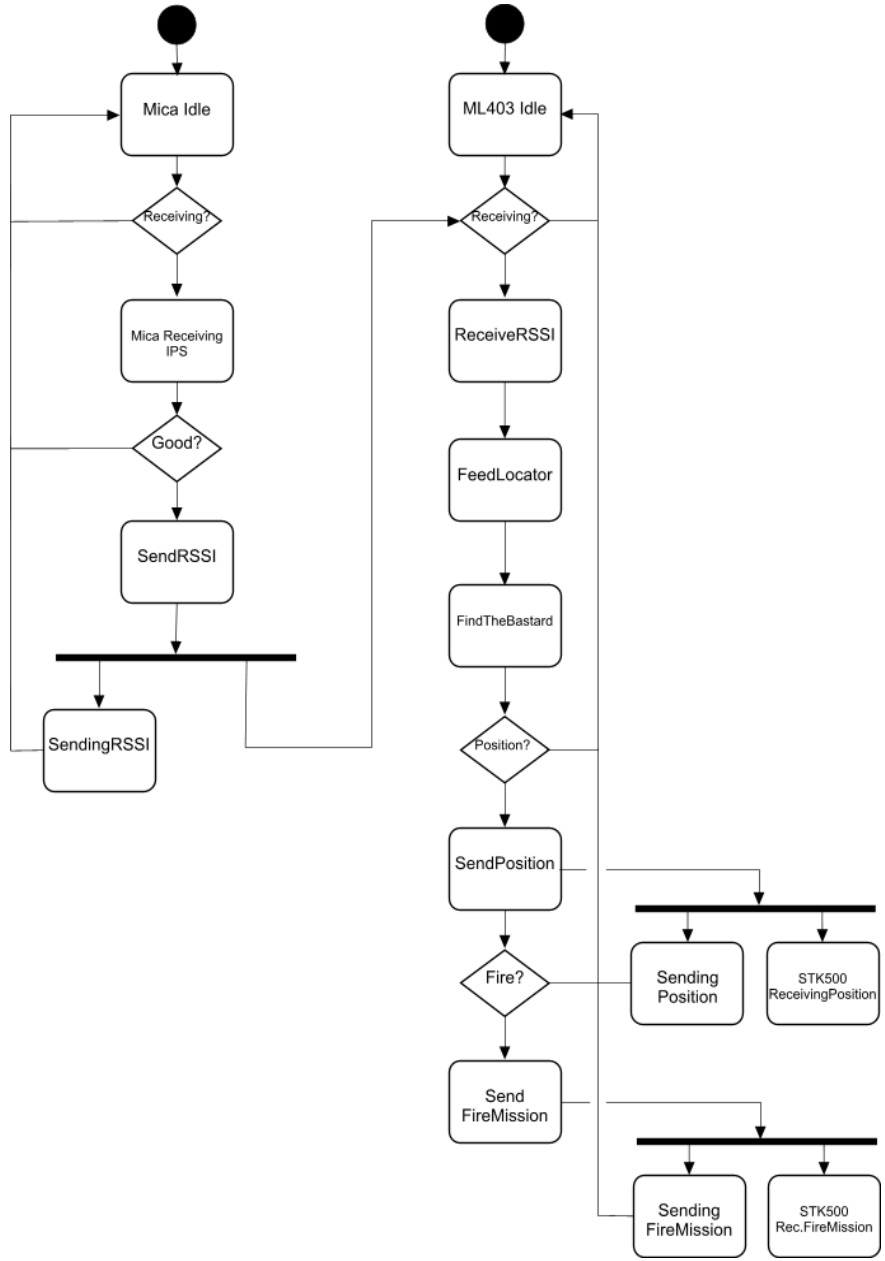
Esse autômato representa a primeira tentativa de projeto do sistema.

Bem precário.

Nesse momento a idéia eram threads independendentes pra tudo.

Projeto

Evolução



Algo espirituoso aqui !

" Espaço deixado intencionalmente em branco

Projeto

Caso de Uso: Mira e Tiro bem sucedido

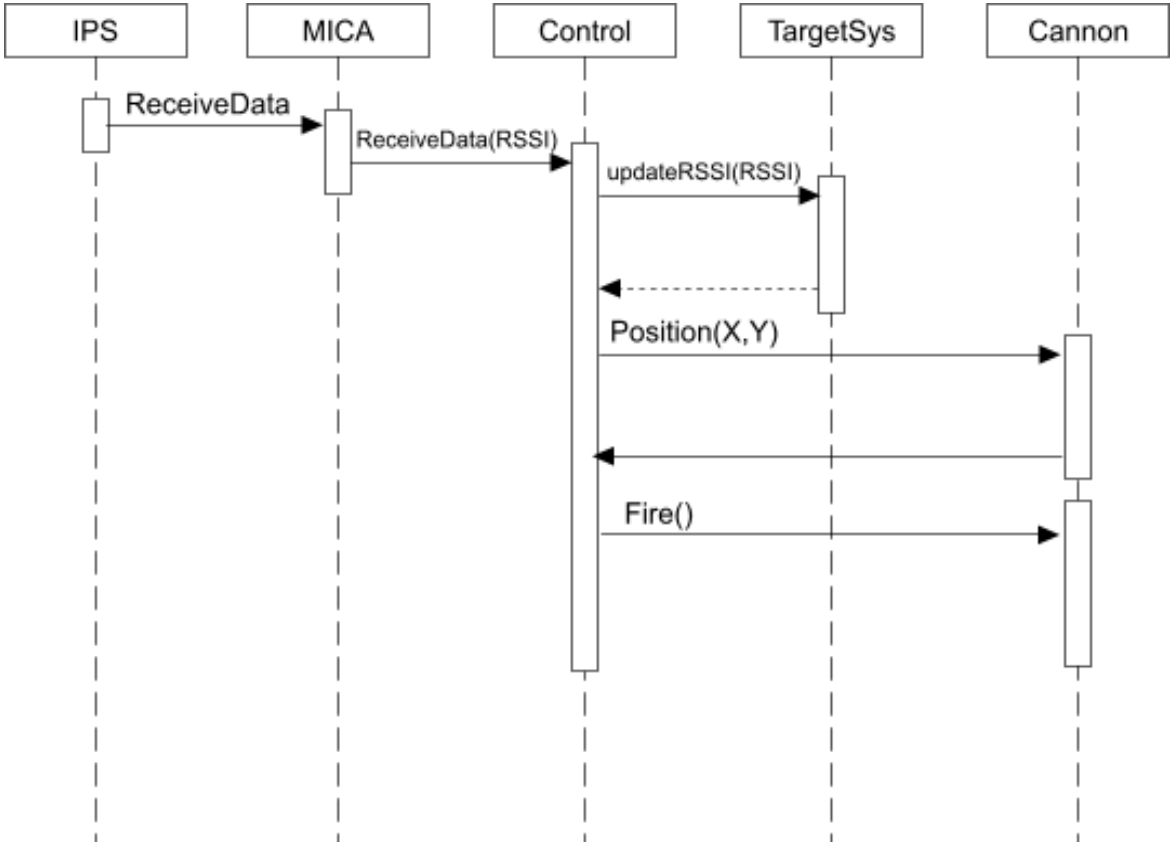
1. O Mica recebe o RSSIData do IPS.
2. Mica envia RSSIData para o Controle.
3. Controle envia RSSIData para o Sistema de Mira.
4. Sistema de Mira tenta definir a Posição do Alvo.
5. Sistema de Mira envia a Posição ao Controle.
6. Controle envia a Posição ao Controlador do Canhão.
7. Controle envia um Comando de Tiro ao Controlador do Canhão.

As variações possíveis envolvem:

- em 4 o Sistema de Mira não consegue definir uma Posição válida
  - Controle não posiciona nem atira o canhão.
- em 4 o Sistema de Mira informa que Posição não é boa o suficiente
  - Controle posiciona, mas não atira o canhão.

Projeto

Caso de Uso: Diagrama



Projeto

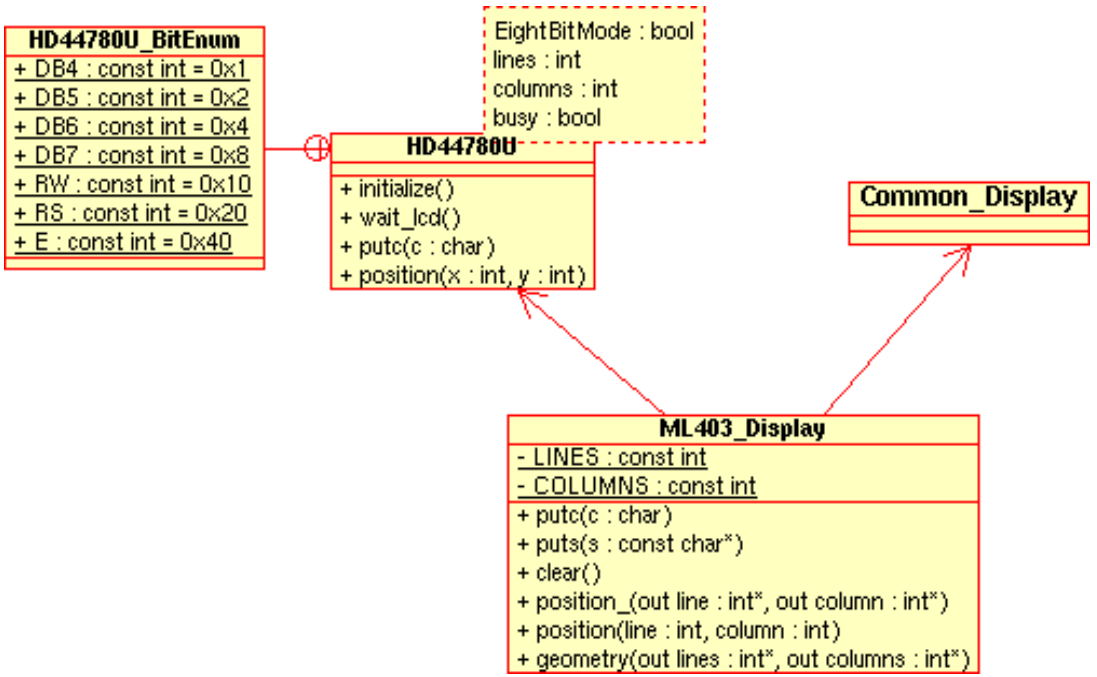
Caso de Uso: Recebe RSSI e Envia UART

1.O Mica recebe o RSSIData do IPS.
2.Verifica se o RSSIData é válido.
3.Envia pela UART para ML403.

As variações possíveis envolvem:
- em 2 se o RSSIData for inválido
  - Volta para 1
- Em 3 envio falhou
  - Envia novamente

Projeto

Display Driver



Implementação

Executivo ciclico na ML403

O seguinte pseudo - código representa o Executivo Ciclico do Controle:

```
boolean hit;

int main() {
    Display display;
    hit = false;
    int attempt = 0;
    while(!hit) {
        PDU = receive _ mica (); // Block !
        Target t = TargetSys . update ( PDU );
        if ( t . fire ) {
            cout << "#" << attempt++
                << " - Fire at [" << t. position . X
                << ", " << t . position . Y << "]\ n ";
            Cannon . position ( p . position ); // Non - Block
            Cannon .fire(); // Non - Block
        }
    }
    return 0;
}
```

Implementação

Display Driver

Após sucessivos fracassos (3 semanas) tentando fazer o display funcionar ele foi dropado , pelo menos por enquanto .

A origem do problema ainda não foi identificada . As tentativas seguiram as orientações das especificações do fabricante , códigos de exemplo e fontes diversas na internet .

Talvez essa responsabilidade passe ao grupo do canhão , que se propôs colocar um conjunto de LEDs BCD para mostrar a posição do disparo .

Uma segunda opção é escrever pela serial para um PC conectado...

Implementação

UARTs

A solução encontrada para as UARTs foi apenas uma mudança no código existente da UART do ML 310...

```
template <unsigned int INDEX =0>
class ML 310_ UART : protected UART_Common
```

e no config .h:

```
typedef ML 310_UART <0> UART ;
typedef ML 310_UART <1> UART_1;
typedef ML 310_UART <2> UART_2;
```

e em mais alguns lugares (traits , types , inits ...).
Isso compila , mas não foi testado .

Implementação

MICAs

Uma estrutura mais elaborada com um formato próximo a isso...

```
int main() {
    ...
    while(1) {
    ...
    if (pdu_rcvd)
        write_uart();
    ...
    }
    ...
}
```

Ainda não foram realizados testes...

Status

- Sem LCD ainda
- Sem as UARTs
- Nenhuma integração
- Fora do planejamento
- *Nada concreto para apresentar.*

Game Over