Ethernet-lwIP-HTTP

Periféricos envolvidos :

- Power Managment Controller (PMC)
- USART
- Ethernet MAC (GMAC)
- TimerConter 0 (TC0)

APIs :

- A Lightweight TCP/IP stack (lwIP)

Resumo :

Esse exemplo demonstra a conexão Ethernet (cabeada) do kit de desenvolvimento SAME-70, criando um servidor web no microcontrolador onde um segundo dispositivo pode acessar a página.

Diagrama

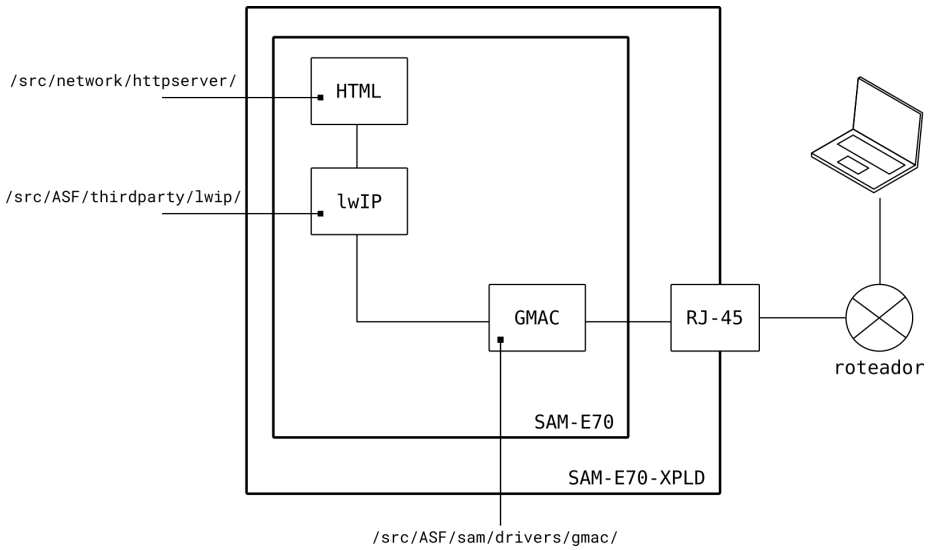

Figure 1: Diagrama de blocos

USART

- BaudRate : 115200
- StopBit : 1 bit
- Paridade : 0 bit
- Data : 8 bits

Utilizado como debug do programa, deve-se utilizar um terminal (exe. putty) no computador para acessar o printf realizado no firmware.

Ethernet MAC (GMAC)

Periférico do uC que implementa uma comunicação 10/100 Mbps Ethernet Mac compatível com padrão IEEE 802.3 (ethernet) podendo operar de forma half ou full-duplex.

O GMAC implementa a camada de **Link** do modelo de camadas da internet, as demais camadas são de responsabilidade do firmware (lwIP).

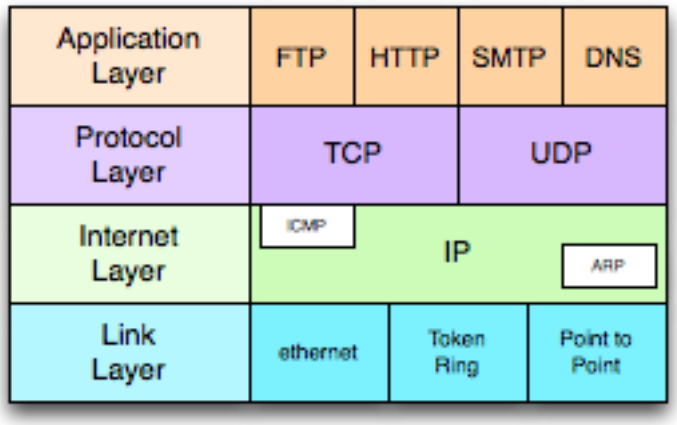

Figure 2: Modelo de camadas Internet [1]

[1] http://network-development.blogspot.com.br/2014/02/layers-of-internet-protocol-suite.html

lwIP

O lwIP é um framework escrito em C focado em embarcados, ele é responsável por implementar e gerenciar as demais camadas do modelo da internet, o lwIP é modular e a lista de módulos é configurada em :

/src/config/lwipopts.h

TimerCounter 0

O TimerCounter é utilizado em modo astável para incrementar uma variável global usada como tick do sistema, é implementado em :

/src/network/timer_mgt.c

```
/**
 * TC0 Interrupt handler.
 */
void TC0_Handler(void)
{
    /* Remove warnings. */
    volatile uint32_t ul_dummy;

    /* Clear status bit to acknowledge interrupt. */
    ul_dummy = TC0->TC_CHANNEL[0].TC_SR;

    /* Increase tick. */
    gs_ul_clk_tick++;
}
```

Executando

1. conecte o kit de desenvolvimento em um roteador via cabo de rede
2. programe o microcontrolador com o código
3. abra a interface serial (COM) para receber dados de debug
4. espere o lwIP buscar um IP no roteador (DHCP) o IP irá aparecer no terminal
5. conecte o computador no mesmo roteador que o kit
6. abra o navegador e acesse o IP do kit (http://IP)

Console

Modificando

Para modificar a página da web basta inserir o arquivo .html em uma constante (/src/network/httpserver/fsdata.c) depois criar uma lista ligada que conecta todos os aquivos desse servidor via a declaração de uma estrutura (final do fsdata.c)

Exemplo :

```
unsigned char data_insper_html[] = "<!DOCTYPE html> <html> <body> <img src=\"./img/logo.png\
```

Depois é necessário criar a estrutura para cada arquivo do tipo fsdata_file:

```
struct fsdata_file {
  const struct fsdata_file *next;
  const unsigned char *name;
  const unsigned char *data;
  const int len;
};
```

Essa estrutura tem 4 conteúdos :

- *next : um ponteiro para a próxima estrutura
- *name : o caminho desse arquivo, exemplo "/img/logo.png"
- *data : o endereço do arquivo criado anteriormente
- len : o tamanho do arquivo

O exemplo a seguir ilustra 5 arquivos com a estrutura a seguir :

```
/
 index.html
 404.html
 insper.html
 img/
   logo.png
   sics.gif
```

```
const struct fsdata_file file_img_logo_png[] = {{NULL, "/img/logo.png", data_img_logo_png +
const struct fsdata_file file_insper_html[] = {{file_img_logo_png, "/insper.html", data_insp
const struct fsdata_file file_img_sics_gif[] = {{file_insper_html, "/img/sics.gif", data_img
const struct fsdata_file file_404_html[] = {{file_img_sics_gif, "/404.html", data_404_html,
const struct fsdata_file file_index_html[] = {{file_404_html, "/index.html", data_index_html
```

Repare que essas estruturas são uma lista ligada sendo que o primeiro elemento da lista é o file_index_html[] e o último o file_img_logo_png