Desenvolvimento de uma Aplicação de Controle de Créditos Usando a Tecnologia *MIFARE*

Julho 2008 – Davi, Marcelo, Murilo, Pablo, Rodrigo

Resumo

MIFARE é uma tecnologia de *smart cards* sem contato, especificada pelo padrão ISO14443. Os *smart cards* servem como um mecanismo simples e barato de identificação e por isso estão na realidade de mercado. Neste trabalho foi realizado um estudo de caso envolvendo um dispositivo *MIFARE* com o desenvolvimento de uma aplicação fictícia de controle de crédito. Diversas dificuldades foram encontradas durante o desenvolvimento de forma que os resultados esperados não foram totalmente obtidos.
Palavras-Chaves: *Smart Cards*; *MIFARE*; *ColdFire*;

1 INTRODUÇÃO

A aplicação desenvolvida implementa um controle de créditos simples, no qual os usuários de um serviço são identificados por seus cartões. Cada usuário pode comprar um certo número de créditos para poder acessar o serviço prestado. Nos pontos de acesso ao serviço, o usuário apresenta o seu cartão de identificação e seus créditos são decrementados, se possuir algum. Sistemas de controle de crédito como esse são muito usados por empresas de transporte coletivo, pois são mais seguros e mais baratos que o modelo de pagamento tradicional. É claro que em sistemas reais há uma preocupação muito maior com questões de segurança, porém o objetivo deste trabalho é realizar um estudo aplicado dos dispositivos usados e não desenvolver uma aplicação comercial completa.

A arquitetura inicialmente planejada para a aplicação está descrita na Figura 1. Nessa arquitetura, o Servidor *Web*, executado sobre um computador pessoal (PC), é o responsável por manter os dados dos clientes do serviço, o identificador do cartão e o número de créditos associado. Esse servidor responde a requisições HTTP para adicionar ou consumir créditos de um dado cliente. Os Clientes de Crédito e de Débito são as aplicações responsáveis por fazer tais tipos requisições, respectivamente. Inicialmente, como está demonstrado na Figura 1, o objetivo era executar tais aplicações em uma plataforma embarcada, a CML5485. No entanto, severas dificuldades foram encontradas no decorrer do desenvolvimento, e a plataforma embarcada foi substituída por um outro PC. As

características gerais da CML5485 e as dificuldades encontradas no desenvolvimento para essa plataforma serão discutidos na Seção 2.

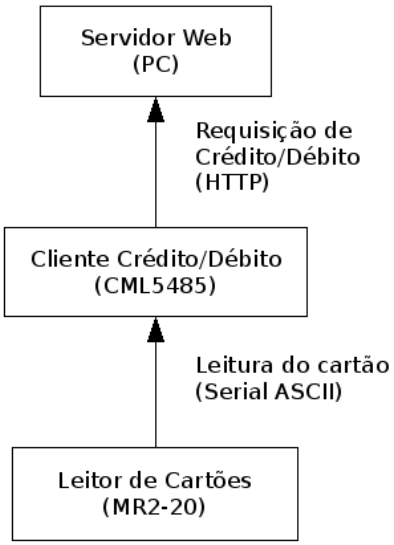

FIGURA 1: Arquitetura Original da Aplicação

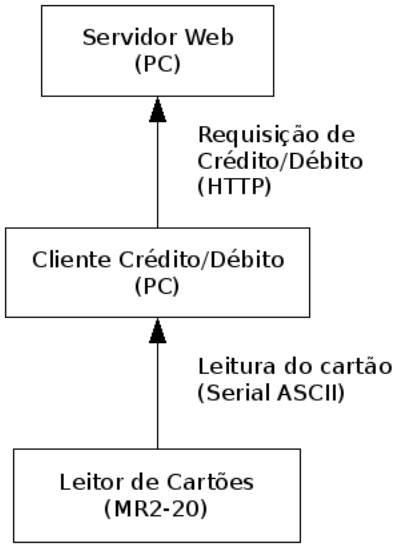

FIGURA 2: Arquitetura Final da Aplicação

A última parte da aplicação é composta pelo Leitor de Cartões. Para essa função foi utilizado um dispositivo *MIFARE* de modelo MR2-20[1]. Esse dispositivo se comunica com

[1] Alguma informação sobre o dispositivo foi encontrada em http://www.albionelectronica.com/a_targetas.htm, na seção *proximity/Readers 13,56MHz/MR2-20*

as aplicações clientes por uma interface serial e envia o identificador do cartão lido. Informações sobre o dispositivo e o protocolo de comunicação serão detalhadas na Seção 3.

2 A PLATAFORMA CML5485

A plataforma CML5485 [CML5485UM], desenvolvida pela *Axiom Manufacturing*[2], é um *Kit* de desenvolvimento para o micro controlador MCF5485 [MCF5485RM], da família *ColdFire* [COLDFIRE], desenvolvido pela *freescale semiconductor*[3]. Essa plataforma possui diversas interfaces de comunicação, incluindo portas seriais padrão e interfaces *Ethernet*.

No planejamento inicial da aplicação de controle de crédito, a plataforma CML5485 hospedaria as aplicações cliente e se comunicaria com o leitor de cartões e com o servidor usando, respectivamente, a porta serial e a interface *Ethernet*. No entanto, graças às dificuldades encontradas com relação a instalação e execução de um sistema operacional nessa plataforma, ela foi substituída por um PC.

1.1 Dificuldades Encontradas para Instalar e Executar Sistema Operacional

O primeiro passo realizado com o objetivo de montar um ambiente de programação para a plataforma CML5485 foi a instalação de uma versão para teste da plataforma de desenvolvimento *CodeWarrior*, desenvolvida pela *freescale*. Tal ambiente contém *cross-compilers* para a plataforma *ColdFire*.

Depois disso, a tarefa foi encontrar um *port* de um sistema operacional para o processador MCF5485. Primeiro apenas foram encontradas distribuições de *uCLinux* em código fonte. Várias tentativas de compilar esse sistema foram realizadas, porém nenhuma foi bem sucedida, provavelmente por problemas de versão ou de configuração do *cross-compiler* utilizado. Percebeu-se, então, que seria necessário encontrar sistemas já compilados e prontos para ser instalados na plataforma. A solução parecia ter vindo com a descoberta de um pacote de suporte (*Linux Board Support Package* – Linux BSP[4]) da

[2] http://www.axman.com/
[3] http://www.freescale.com/
[4] http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MCF548X&fpsp=1&tab=Design_Tools_Tab

freescale para a plataforma em questão. O BSP contém imagens do *kernel linux* e diversos utilitários compilados para o processador em questão.

O primeiro passo para instalar o sistema na plataforma é iniciar uma conexão serial do computador usado para configurar a plataforma (*host*) e a plataforma em si (*target*). Essa conexão pode ser realizada por qualquer programa como o *minicom* ou *cu*. Por padrão, a plataforma possui, na seção de *boot* da *flash*, o *bootloader* dBUG. Esse *bootloader* possui diversas funcionalidades, inclusive permite o *download*, via TFTP, de arquivos para a memória da plataforma e acessar a *flash*. Essa função foi usada para permitir a instalaçãode outro *bootloader* mais apropriado para iniciar o sistema Linux, o *colilo*.

Para incorporar o colilo à plataforma, primeiramente foi necessário criar um servidor TFTP. Para isso foi usado o *atftpd*, e foi preparado com o *bootloader* e as imagens. Este servidor será utilizado também para as outras fases.

Depois de criar o servidor TFTP, deve-se conectar a plataforma à rede, de forma que possa acessar o servidor. Depois disso, configura-se, por meio da interface dBUG, as informações necessárias para realizar o *download* dos arquivos. Isso foi feito usando os comandos abaixo:

```
DBUG> set server 192.162.1.1
DBUG> set client 192.162.1.2
DBUG> set gateway 192.162.1.1
DBUG> set netmask 255.255.255.0
DBUG> set filetype srec
DBUG> set filename colilo_mcf5485.srec
```

Esses comandos configuram o acesso à rede e as informações do arquivo a ser copiado. O comando abaixo inicia o download:

```
DBUG> dn
```

Depois disso o *colilo* já se contrará na memória e poderá ser executado diretamente. O comando abaixo faz isso:

```
DBUG> go 1000400
```

Com o colilo executando, resta ainda carregar as imagens do sistema para a memória e configurar a partição de *root* do sistema. Para carregar as imagens usa-se procedimento semelhante ao que serviu para carregar o *colilo*. Para configurar a partição de *root*, o recomendado no manual do BSP é usar o servidor NFS. A maior dificuldade encontrada foi nesse momento, pois o grupo não conseguiu montar, a tempo, a infra-estrutura necessária.

3 LEITOR DE CARTÕES

O leitor de cartões do sistema de controle de crédito foi implementado por um dispositivo de leitura por proximidade para cartões *MIFARE* MF1 e *MIFARE UltraLight*. O dispositivo possui uma interface de comunicação serial que pode operar em três modos distintos: *Wiegand 26 bits*, *MSR ABA Track 2* e RS232. Para a aplicação desenvolvida foi utilizado o modo RS232 que usa um protocolo ASCII simples, ilustrado na Figura 3.

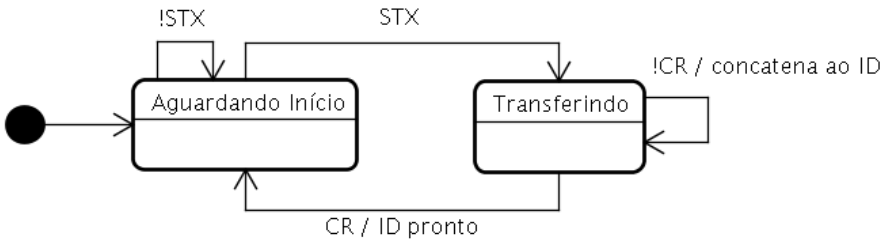

FIGURA 3: Protocolo de comunicação com o leitor de cartões

O caractere ASCII STX (0x02) sinaliza o início de uma transferência. Depois disso, o dispositivo envia os caracteres que compõem o identificador do cartão. A transferência é finalizada com o caractere CR (0x0D). É muito provável que o protocolo insira caracteres de escape (ESC) caso os caracteres de controle ocorram no identificador do cartão, no entanto isso não ocorreu com nenhum dos cartões usados para teste e esse caso não foi tratado nas aplicações cliente.

Para que a interface de comunicação do MR2-20 funcione em modo Serial ASCII, as seguintes conexões físicas, dos pinos da interface DB9, devem ser feitas: o pino 5 (*signal ground*) do receptor deve ser conectado ao *ground* da alimentação do MR2-20 (fio preto) e o pino 3 (*transmitted data*) deve ser conectado ao sinal de dados do MR2-20 (fio verde).

4 REFERÊNCIAS

[CML5485UM] *Application Board for Freescale MCF5485 MCU Hardware User Manual*. Axiom Manufacturing, 2005.

[MCF5485RM] *MCF548x Reference Manual*. freescale semiconductor, 2006. Disponível em: <http://www.freescale.com/files/32bit/doc/ref_manual/MCF5485RM.pdf>.

[COLDFIRE] *ColdFire Family Programmer's Reference Manual*. freescale semiconductor, 2005. Disponível em: <http://www.freescale.com/files/dsp/doc/ref_manual/CFPRM.pdf>.