SMALL PROTOCOL
README SAMPLECODE

eDIP-SERIES - FAST AND EASY CONNECT TO A MICROCONTROLLER

INTRODUCTION

You will find a short example to implement SmallProtocol in C-Language into a microcontroller.
The code is just an example, just to show how the the protocol works. It should be adapted to the specific needs.
The code is only tested on Renesascontroller R8CM12A.
The code is supplied by ELECTRONIC ASSEMBLY and provided „as is“.
ELECTRONIC ASSEMBLY makes no warranties regarding function.

OVERVIEW SMALLPROTOCOLL

Every eDIP woks ia a similar way, looking at SmallProtocol, further information you can find in the data sheets (http://www.lcd-module.de/datenblaetter.html). Following is just a short overview:
Each data transfer is embedded in a fixed frame with a check-sum (protocol package). The EA eDIP acknowledges this package with the character <ACK> (=$06) on successful receipt or <NAK> (=$15) in the event of an incorrect check-sum or receive buffer overflow. In the case of <NAK>, the entire package is rejected and must be sent again (refer to Fig. 1).
In Fig. 2 you will find the transformation of the theoretic protocol into the example-code by means of switching off terminal (#TA).

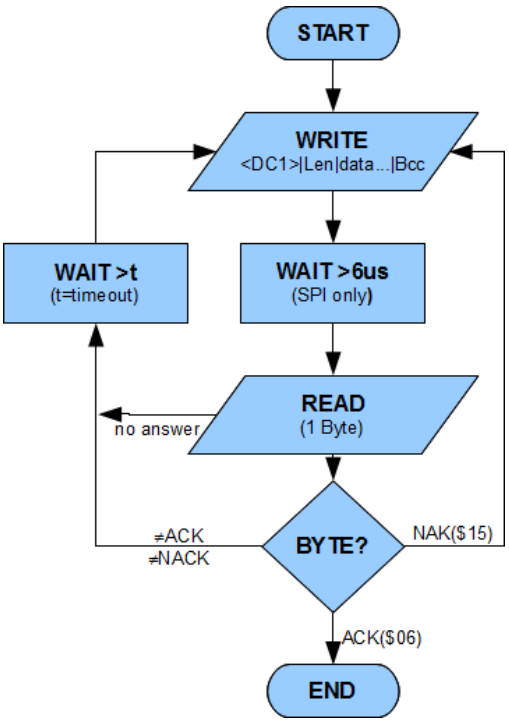

fig. 1: Overview SmallProtocol

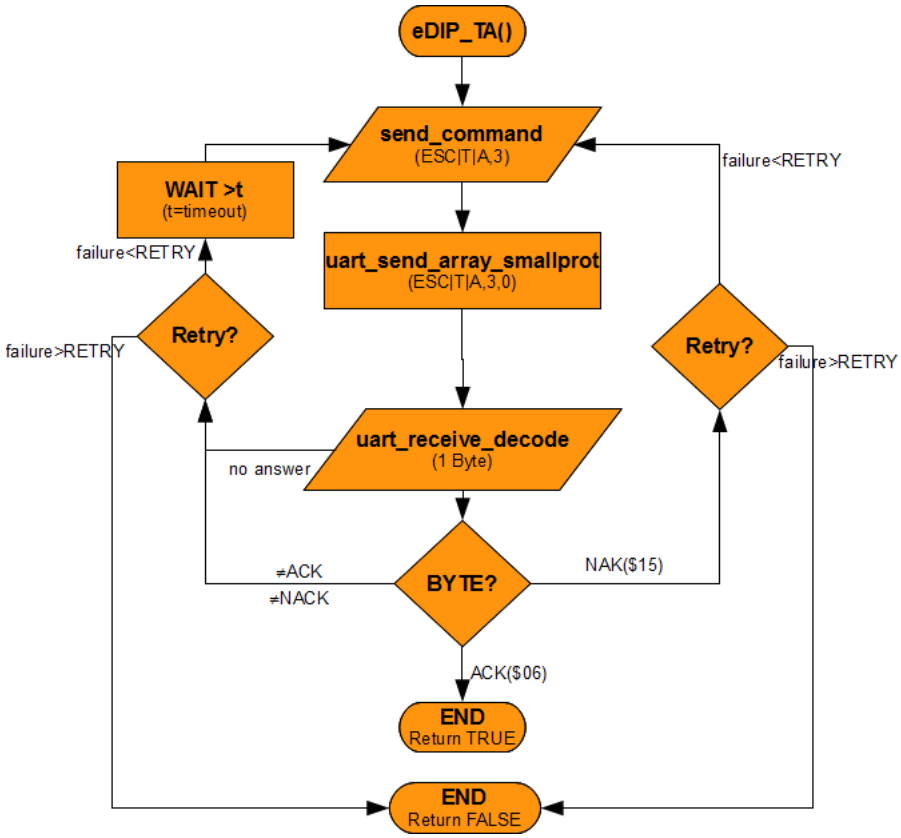

fig. 2. SmallProtocol transfered in C-Code

REQUEST BUFFER CONTENT WITH SMALL PROTOCOL

The protocol includes also a possibility to get data from the send buffer of the display. Because of not using hardware RTS/CTS the microcontroller has to request the buffer contents.
There are some ways to detect that bytes are ready in the send buffer of the display:

- Monitoring pin 20 (SBUF), it goes low if data is present. If you detect data is available, you can request the buffer content.
- Polling buffer information. You request the buffer information in a fixed time slice. If you detect some send bytes are ready, you can request the buffer content.
- Polling buffer. It's nearly the same as above, but you call ''request buffer content". If there is no byte ready to send, you get ''<DC1>|0|17" back.

We chose possiblity three for our example. In fig.3 you can see the sequence of operations in general, fig. 4 shows the impleted structure.

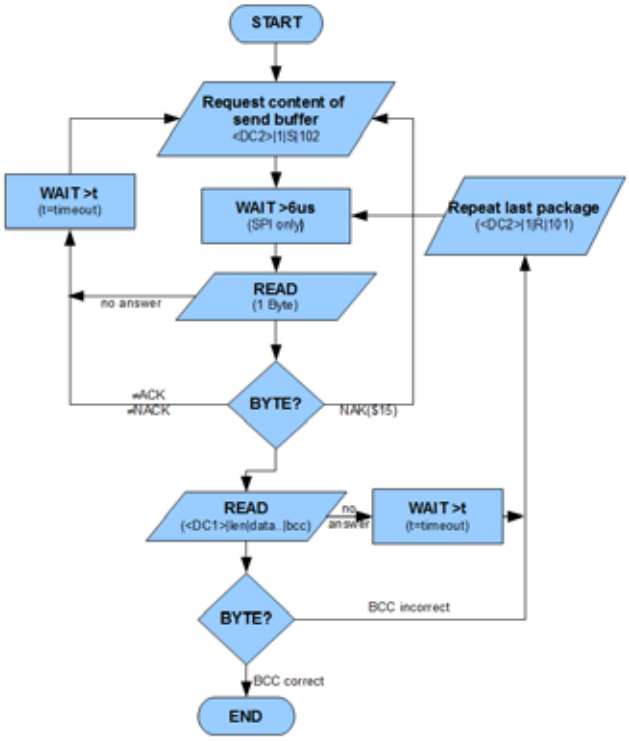

fig. 3: Overview Request buffer content

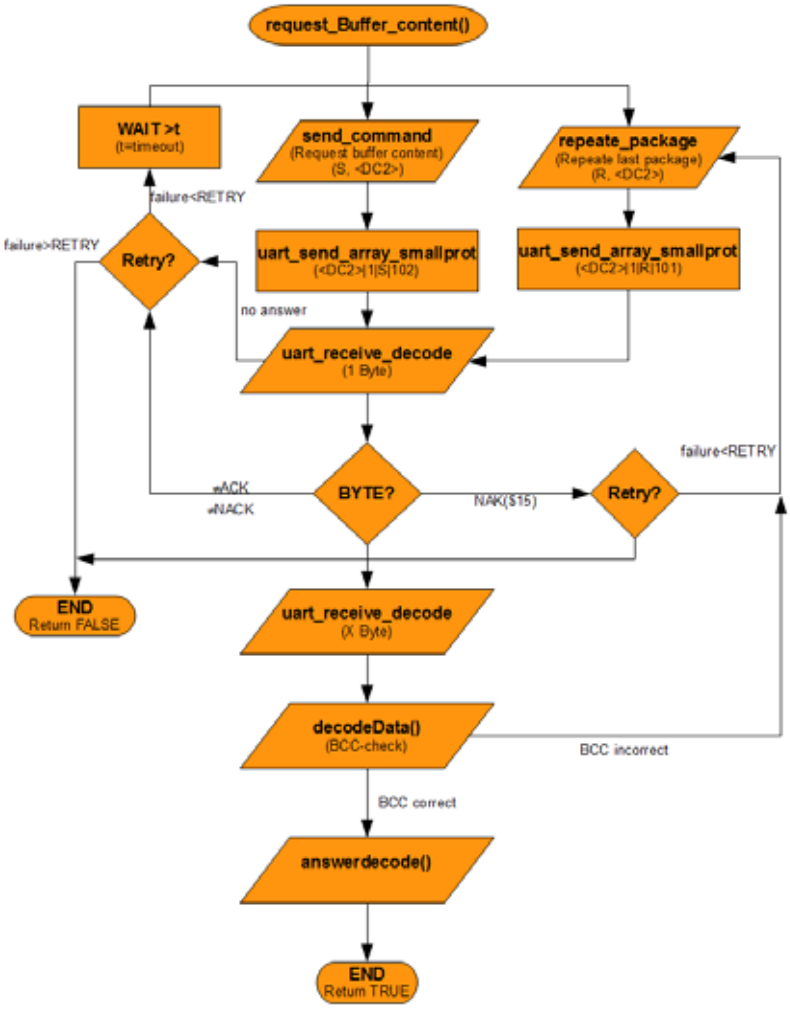

fig. 4: Implemented code to request buffer content

Former informations about the SmallPrtotcol you will find in the datasheets of the eDIP-Displays. You will find the datasheets under:
http://www.lcd-module.com/datasheets.html

THE C-CODE IN DETAIL

The project is split up into 4 modules: main, timer, uart and display. All modules consist of two files, a header-file and a c-source-file.
Main-module is used as general control. It just calls the functions of the other modules.
Timer-module is only used to provide a timer with 1ms.
Uart-module provides the basic communication functions between micro-controller and eDIP.
Display-module uses uart-module to provide display commands to user.

Main-module

This module calls the initialization of hardware, like timer, interface, I/O-ports and frequency. In addition you will find the call of a sample screen and the main-loop.
Printing and typographical errors reserverd

Timer-module

Includes initialization of timer and interrupt handling of the 1ms-timer.

Uart-Module

This module includes basic communication functions. You will find general defines concerning serial interface here, like size of Rx-, Tx-buffer, timeout, retry (how often a package is send again) and time to poll for new bytes in send buffer of eDIP.
One of the most important function is uart_send_array_smallprot, which provides the SmallProtocol around inputed data.
The second function you should take care of is uart_receive_decode, which monitors the answer of the eDIP. It checks the incoming data.

Display-Module

In this module you will find many commands to control the eDIP. You can look in the data sheets at the section „All commands at a glance“ and use them nearly the same way.
You will find a function, that reacts on answers of the display, like touchbuttons.