Additional information about ARM code

Albert Dorn

11. Dezember 2006

Inhaltsverzeichnis

1 Debugger-Module

The debugger exists of one header and C file. For using the debugger in a already working ARM project you just need to include the header file in the main class. Make sure that your project has access to the C debugger file. The debugger is based on an interrupt which starts with each access to the UART1 port. The communication logic works with a charset of ASCII letters where some of them have special functions (see Product Description for a list of those letters). The debug messages are a charset with an end of line information also called String which are consitst of (information = value). If you want that E.R.D.E. shows debugging status information you need to be sure the output stream consist of (information = value). For more details take a look at the well documented code especially the ISR method.

The debugger module is subject to following restrictions.

- You need to make sure the project you include the debugger module is not unsing the UART1. Because the whole communication with the E.R.D.E. is done using the UART1.
- The debugger freezes when you include it into the Clown program of the thouchpad. Tests have shown that only when you enter the debug mode while the Clown programm is running, the whole programm freezes. After the Clown programm does its whole animations you are able to debug the program. This may result because both ISRs are using the same registers of the ER. Solving appendages like using a mutex for the problematical registers where not successful.

2 ER code Uploader

The code upload implementation on the ARM side exists of a 512 bytes fifo buffer. When you start the code uploader in E.R.D.E. ARM first recieves the whole code in a bytearray. E.R.D.E. always sends 512 bytes even if the code part is much less. After this byteharray

is full the ARM initiates the logic which sends the whole code to the E.R.. We could not test the part between the ARM and the E.R. program register due to the hardware failure in the FPGA. Following flow charts diagramm <flowchart uploader> shows the logic of the ER code uploader. The red part is partly implemented but not tested.

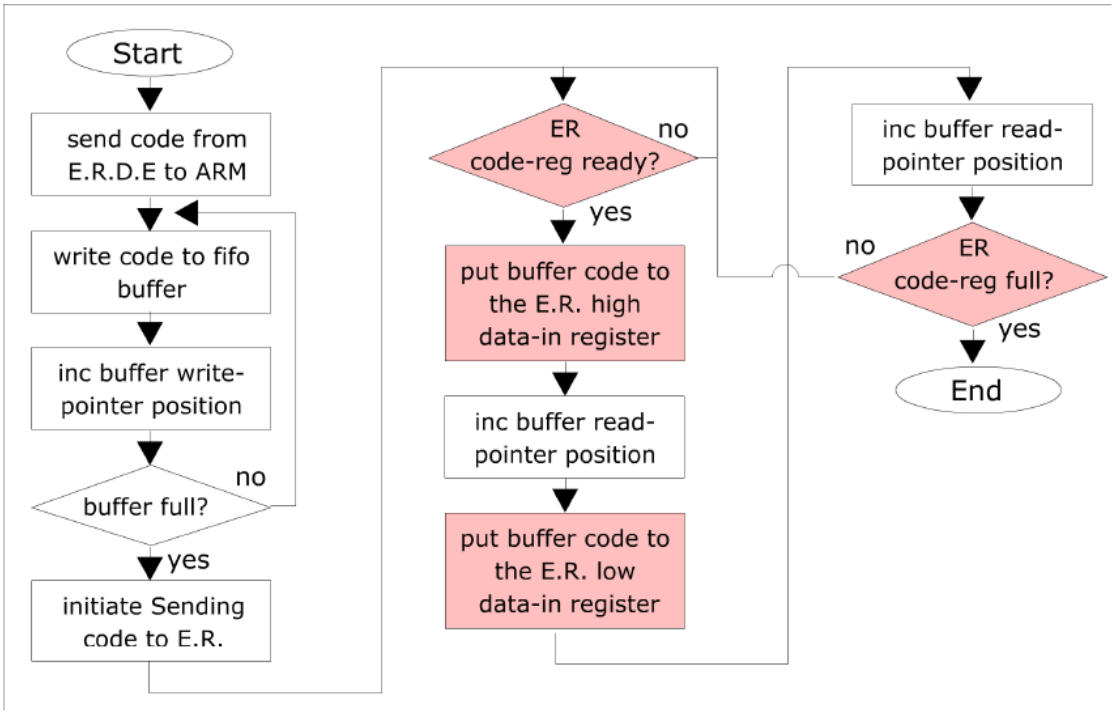

Abbildung 1: Flowchart code uploader