E.H. HOVHANNISYAN

A METHOD FOR ASSESSING THE EDUCATION OUTCOMES BASED ON THE RUSH MODEL

The possibility of quantitative assessment of education outcomes based on the Rush model is studied. A method for obtaining the assessment taking into account the threshold values of weighted coefficients of the education outcomes' components has been proposed.

Keywords: Rush model, education outcome, outcome components, weighted coefficient.

UDC 681.326:519.713

N.M. NERSESYAN

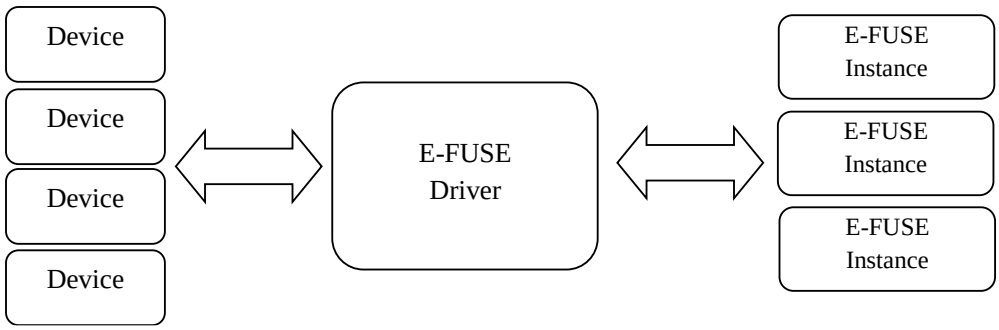
INTERACTION INTERFACE OF AN ELECTRICAL FUSE DRIVER

For Electrical One Time Programmable Memories (EFUSE) interconnections IEEE 1500 standard is chosen with mandatory BYPASS and STATUS_SEL instructions. The control of FSM and EFUSE by IEEE1500 interface are explained.

Keywords: EFUSE automation, EFUSE interface, EFUSE driver.

1. Introduction.

Before automating the system of generation E-FUSE drivers, we must establish I/O connections with the outer world. A driver can be automatically adjusted to custom E-FUSE which has no common I/O pins. But interaction with other devices and drivers must be established constant and cannot be changed from project to project:



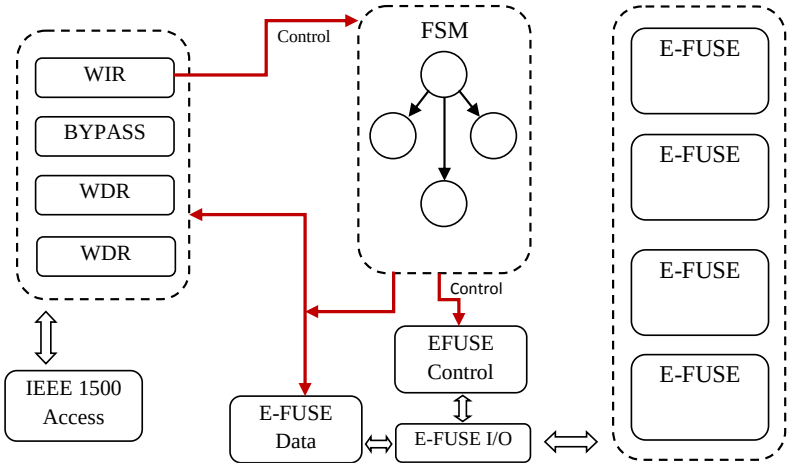
2. Driver I/O interface.

Because the E-FUSE system is part of an embedded system, IEEE 1500 interface is chosen. First of all, this interface is so flexible that can support our needs, and second, it has only few mandatory rules that we must include in the system, most rules are optional and we can ignore them.

Thus, the driver module will contain IEEE 1500 mandatory WIR and BYPASS registers, mandatory I/O pins (SelectWIR, ShiftWR, CaptureWR, UpdateWR, WSI, WSO, WRCK, WRSTN). This will provide a connection with external devices by the IEEE 1500 interface.

The driver module will also contain E-FUSE instances I/O pins (custom for each E-FUSE type).

The figure below illustrates the logic scheme of the system:



As we can see from the figure, E-FUSE I/O pins are divided into 2 categories: E-FUSE Data and E-FUSE Control. FSM (Finite State Machine) controls the values of E-FUSE Control pins and the direction of the E-FUSE data. FSM cases are controlled by the WIR register (IEEE 1500 instructions). The WIR register values describe the user in the technical file. Thus, we can see the flow from the technical file to the direct control of E-FUSE instances.

3. IEEE1500 registers and instructions.

All tasks in the technical file will become instructions after creating the driver module. Those instructions will be accessible by the IEEE1500 standard. Tasks in the technical file describe the values of E-FUSE pins at each time step. That description becomes the case flow in FSM. And, when the task-instruction is asserted on the WIR register the driver module runs FSM and the implemented task flows on the E-FUSE module. Thus, WIR register is used to implement tasks from technical file (E-FUSE description file). The size of the WIR register depends on the tasks` count in the technical file.

The BYPASS instruction is a mandatory instruction in the IEEE1500 standard. It is used when there is no need to interact with the driver. It just connects 1bit register, which shifts out the information shifted in.

The STATUS_SEL instruction selects the status register between the WSO and WSI pins. The status register contains information about the FSM ready state and fails during processing tasks.

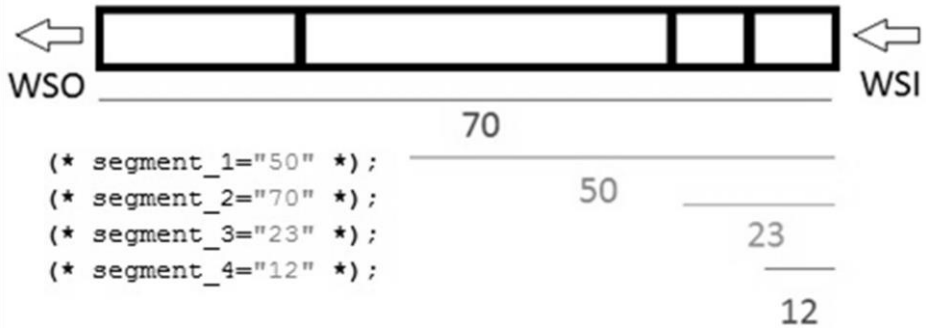
The SPACE_SEL instruction selects the space number register between WSO and WSI pins. The Space number register contains the number of currently active spaces of E-FUSE. The user can work only with a currently active space. The number of spaces must be given in the technical file.

Each task described in the technical file must select or create a self-chain connected between WSI and WSO. If the chain in the task is not selected, the status register will be selected by default.

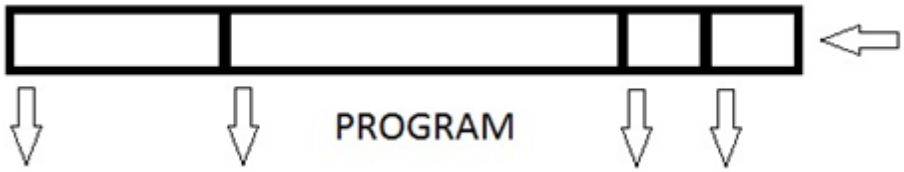
4. Information buffering.

The purpose of E-Fuse is to store information that the user can read later. Thus, the user must prepare information somewhere and give a task to the E-Fuse driver to store that information in the E-Fuse instance. For this purpose, in the E-Fuse driver, we create a **buffer register,** which can be selected by BUFFER_SEL instruction and will create a chain between WSI and WSO.

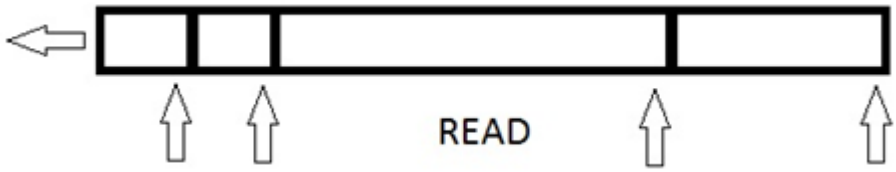
Any information that the user wants to write into the E-Fuse instance must be shifted into the buffer register. If there are several segments of information each segment will be written separately. The buffer register size will be equal to that of the segment which has the maximal size. The rule of the IEEE 1500 Standard is to have 1 chain for 1 instruction, thus, we can only have 1 buffer register with a maximum size, which can support smaller segments, too:



Thus, for programming we must take information from the LSB part if the buffer does not contain a full-sized segment:



For the reading operation, things go otherwise, because the WSO pin is connected to the MSB part, information from the E-FUSE instance must be placed from MSB to LSB in the buffer register:



This solution makes the driver flexible and does not break the IEEE 1500 rule on 1 instruction for 1 chain.

5. A driver as a part of embedded system.

With a dedicated interface, the E-FUSE instance depends only on the effuse driver module, but not on the embedded system of other modules of connections. Thus, the embedded system becomes independent of the EFUSE type, which is a part of automation standard.

6. Conclusions.

Today, E-FUSEs are strongly demanded. Thus, it is important to create a control module for E-FUSE easily and quickly. The automation process of the E-FUSE drivers requires a standard for outer world interconnections. Because E-FUSEs are generally integrated in SOC systems, IEEE 1500 standard has been chosen. The task concept in the technical file makes the IEEE 1500 instructions flexible: the user can define any instruction he wants and describe its behavior in the technical file. For information buffering, one big buffer register concept is chosen. The WSO and WSI chain is connected by the buffer register and all read/write information for each segment is shifted in/out by the buffer register with only one BUFFER_SEL instruction. Thus, the E-FUSE driver consists of 2 separate parts: the IEEE 1500 interface driver and FSM which controls the E-FUSE instance outgoing pins.

REFERENCES

1. **Brent B.** Welch**.** Practical Programming in Tcl, 2004.
2. IEEE 1500 - Standard for Embedded Core Test.
3. Verilog programming guidelines.
4. Verilog Hardware Description Language Reference Manual,1991.
5. E-FUSE documentations.

Ն.Մ. ՆԵՐՍԵՍՅԱՆ

ՄԵԿԱՆԳԱՄՅԱ ԷԼԵԿՏՐԱԿԱՆՈՐԵՆ ԾՐԱԳՐԱՎՈՐՎՈՂ ՀԻՇԱՍԱՐՔԵՐԻ ԴՐԱՅՎԵՐԻ ՓՈԽԱԶԴԵՑՈՒԹՅԱՆ ԻՆՏԵՐՖԵՅՍԸ

Մեկանգամյա էլեկտրականորեն ծրագրավորվող հիշասարքերի դրայվերների համար ընտրվել է ինտերֆեյս: Բացատրվում է ինտերֆեյսի օգտագործման կարգը՝ վերջավոր ավտոմատի և հիշասարքի կառավարման համար:

Առանցքային բառեր. մեկանգամյա էլեկտրականորեն ծրագրավորվող հիշասարք, դրայվեր, ինտերֆեյս:

Н.М. НЕРСЕСЯН

ИНТЕРФЕЙС ВЗАИМОДЕЙСТВИЯ ДРАЙВЕРОВ ЭЛЕКТРИЧЕСКИ ОДНОРАЗОВЫХ ПРОГРАММИРУЕМЫХ ЗАПОМИНАЮЩИХ УСТРОЙСТВ

Выбран интерфейс для драйверов электрически одноразовых программируемых запоминающих устройств. Объясняется использование интерфейса для управления конечным автоматом и запоминающим устройством.

Ключевые слова: электрически одноразовое программируемое запоминающее устройство, драйвер, интерфейс.