

Scalable Simulation Environment of Microcontrollers with Remote Access

Sergei Bykovskii^a, Tatyana Prilutskaya^a and Elizaveta Kormilitsyna^a

^aITMO University, 49 Kronverksky Pr., St. Petersburg, 197101, Russian Federation

Abstract

The article deals with the problem of developing a scalable simulating environment of microcontrollers with remote access. The authors proposed the solution based on Tornado Webserver and SystemC models of hardware that can be executed in parallel. The users can connect to the server remotely, test their microcontroller software and then easily move it to real hardware. Currently, the system supports the programming of STM32F4 series microcontrollers integrated into SDK1.1M educational board. The system has been developed by ITMO University and is available on Open EDU platform based on Open edX. It is used for the teaching of Embedded System course. The average compilation time for one project is 3-7 seconds depending on complexity of the project and the number of users worked in parallel. The proposed system can be scalable on several processors (vertical scaling) and servers (horizontal scaling) to reduce the average access time and to increase the number of parallel users.

Keywords

Microcontrollers, Embedded Systems, Internet of Things, Cyber-Physical Systems, Simulation, Software-in-the-loop, Remote access

1. Introduction

In some cases, it is not able to use a real hardware to test developed software for microcontrollers. The special simulators are needed for this purpose. There are several tools for simulation microcontrollers and its environment with peripheral devices.

Some tools are integrated in IDE for software developers and provide the simulation environment for local projects. The developers should install the IDE and use it on their own computers. The simulation process requires a lot of computing resources because it needs to provide a possibility to test the embedded software in real time. If their own computers is not so powerful the developers can not use this tools.

Other tools are fully accurate and simulate low-level details of the system. This kind of tools can not be used for testing the real-time interaction with control elements such as buttons, switches, touch-displays etc.

Other tools can simulate only the part of microcontroller like processor or processor with

Proceedings of the 12th Majorov International Conference on Software Engineering and Computer Systems, December 10–11, 2020, Online & Saint Petersburg, Russia

✉ sergei_bykovskii@itmo.ru (S. Bykovskii); prilutan@gmail.com (T. Prilutskaya); sholohova.elizaveta@mail.ru (E. Kormilitsyna)

🆔 0000-0003-4163-9743 (S. Bykovskii); 0000-0002-5316-6763 (T. Prilutskaya); 0000-0002-5182-6940 (E. Kormilitsyna)



© 2020 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

such peripheral devices as timers, interrupt controller and can not simulate wide range of peripherals (displays, keyboards, sensors etc).

The authors proposed the architecture of the system that can be scalable on several processors and servers and provide the remote access to the simulation environment. It simulates the hardware on the intermediate level of abstraction (loosely timed) and can provide a real-time reaction on user actions like pressing buttons and others.

2. Related works

In this section, we observe the existing tools that can be used for microcontrollers simulation and their environment. We observe such tools as Keil μ Vision, MATLAB&Simulink, QEMU, Tinkercad and Jumper Virtual Lab.

Keil [1, 2] is an MCU software development environment for projects of any complexity. Among other tools like IDE, project manager, Performance Analyzer and debugger it supports complete instruction set simulation for ARM cores and simulation of the on-chip peripherals.

MATLAB [3, 4] programming environment with Simulink becomes a powerful tool for system-level design and multidomain modeling. Simulink computes a state of the system on each simulation loop phase respective to the current model time step. A model time advances according to the solver program. Simulink supports code verification without target hardware by software-in-the-loop simulation (SIL). Some execution-time metrics could be collected while simulation.

QEMU [5, 6] is a hardware emulator. QEMU translates instructions of the target system into operations in C, compiles them and runs on the host system. It supports parallel emulations, debugging, user and system modes, and much more. A wide list of architectures (see Table 1) and peripherals is implemented. Here's a partial list of them: watchdog, EXTI, ADC, AFIO, SPI, PCI, I²C. Despite its rich functionality, QEMU is unable to provide a sufficiently accurate hardware operation: the program does nothing about the timing of execution, it simply runs the target system code as fast as it can.

Tinkercad [7, 8] 3D modeling service's Circuits extension for electronic circuits simulation provides a set of off-the-shelf electronic components, graphics editor, sensor readings configuration, and collaboration tools. It implements the ability to connect some Tinkercad peripherals like timers, LCD, LED to a virtual MCU via UART, SPI and I²C pins. Tinkercad has clear virtues like visibility, intuitive interface, and a large community being useful in the studying process. But the lack of supported industrial MCUs results in inapplicability for serious projects.

Unlike previous ones, Jumper¹ is a dedicated tool for hardware emulation only. It is being promoted as a solution for CI/CD implementation in embedded systems. Jumper normally provides real-time hardware emulation, acceleration is available. Jumper SDK has Python modules for interaction with virtual MCU via UART and GPIO APIs. There is also Peripheral Modeling Framework providing peripheral pinout and configuration templates for conjugation via SPI or I²C. However, it is up to the user to implement the desired device's behavior.

The comparison of observed tools is presented in table 1.

¹<https://docs.jumper.io/>

Table 1
MCUs simulation tools comparison

The tool	Emulated CPU/MCU or architecture	Environment	Non-commercial distribution	Ease of use
Tinkercad	Arduino UNO, ATtiny85, BBC micro:bit	Browsers	Partly	+
QEMU	ARM, SPARC, MIPS, m68k, AVR, RISC-V, SH-4,	Windows, Linux, macOS	+	-
Jumper Virtual Lab	NUCLEO-F411RE, nRF52832	Docker	+	+
Keil μ Vision	Arm-based MCUs	Windows	Shareware	-
MATLAB&Simulink	STM32, Hercules ARM Cortex-M, ARM Cortex A, ARM Cortex-R	Windows, Linux, macOS	Shareware	-

QEMU emulator provides a fully accurate simulation. It can not be used for real-time simulation because model time is several times slower than real-time. Keil and Matlab&Simulink are local solutions for a simulation. It is hard to scale them and provide remote access for multiple parallel users. Jumper Virtual Lab is more suitable for our purposes but it seems that is out of support now by their developers because the main website of this site is closed.

3. Proposed system

3.1. Architecture

We propose the architecture of the Embedded Systems Virtual Laboratory that is scalable and can provide remote access. We also implement SDK-1.1M² development board model as an example of using the proposed architecture.

The architecture of the virtual laboratory is shown in Fig. 1.

The system has a client-server architecture. Clients interact with the system via a Web browser. Entering the website creates an individual session for each user. During the work, a user is assigned a unique identifier (id₁, id₂, ... id_N). Access control to server resources is realized with help of such an identifier.

On the website, users can upload code for a programmable embedded system, compile an executable file of the System Model and run it for execution. The results of the work will be displayed on the Browser window. The client-side interacts with the server-side via Web sockets. Tornado Web server executes user session on server site. It provides the scalability of the virtual laboratory. Tornado server can dispatch process to the several CPU of servers

²<https://isup.ru/news/14210/>

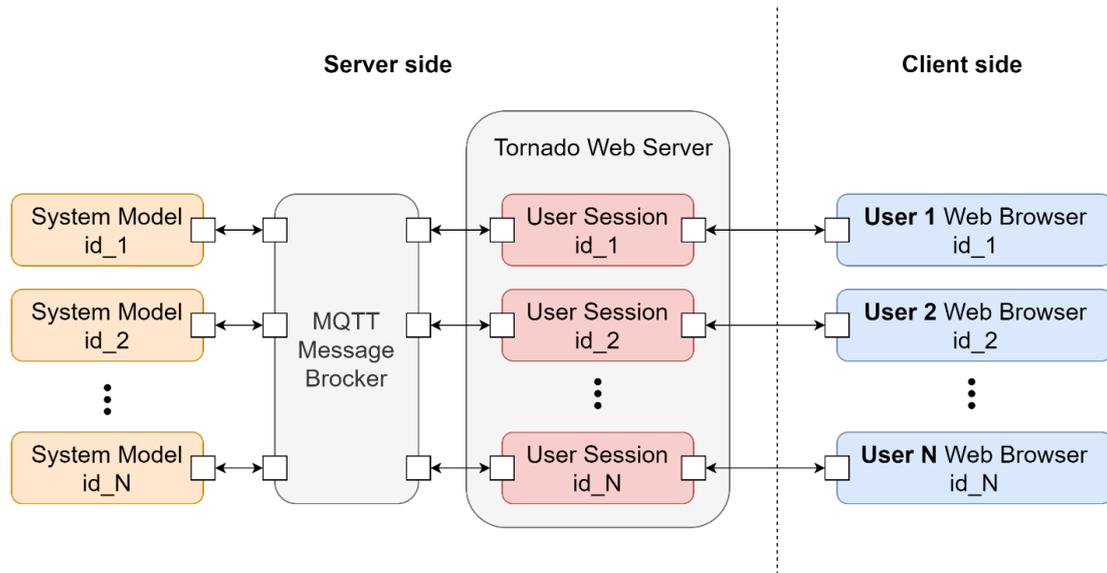


Figure 1: The architecture of virtual laboratory

and can be executed on several servers. Protocol MQTT provides data transfer between user sessions and system models, this transfer is organized through the message broker. Eclipse Mosquitto is used as a message broker.

The system model is a microcontroller which detailed block diagram is shown in Fig. 2. This microcontroller contains one CPU and programmable input/output peripherals. The common bus is used as topology and connects CPU with peripheral devices. As peripheral devices, there are the following units: timers, GPIO Controller, UART, Interrupt Controller, ADC.

The model consists of two parts: the system part and the user part. The system part is the loosely timed TLM model of all system units. The user part is a code that executes on CPU. The way to get an executable model file is shown in Fig. 3. The model was developed using the SystemC library of C++ programming language. These types of models provide the possibility of real-time reactions to user actions.

All interactions are presented as information messages towards client's Web browser. System model communicates with a User session via MQTT broker. A User session is supported by Tornado server. Further, in a User session, all messages are transmitted to Client's browser using Web socket.

The behavior of system peripherals is simulated on the client site in a Web browser.

All data between System model, User sessions and Web browser is transferred in JSON format.

Communication via MQTT broker is provided through two channels for each user in publisher/subscriber mode according to Table 2.

The Fig. 4 is shown how components of virtual laboratory interact with each other.

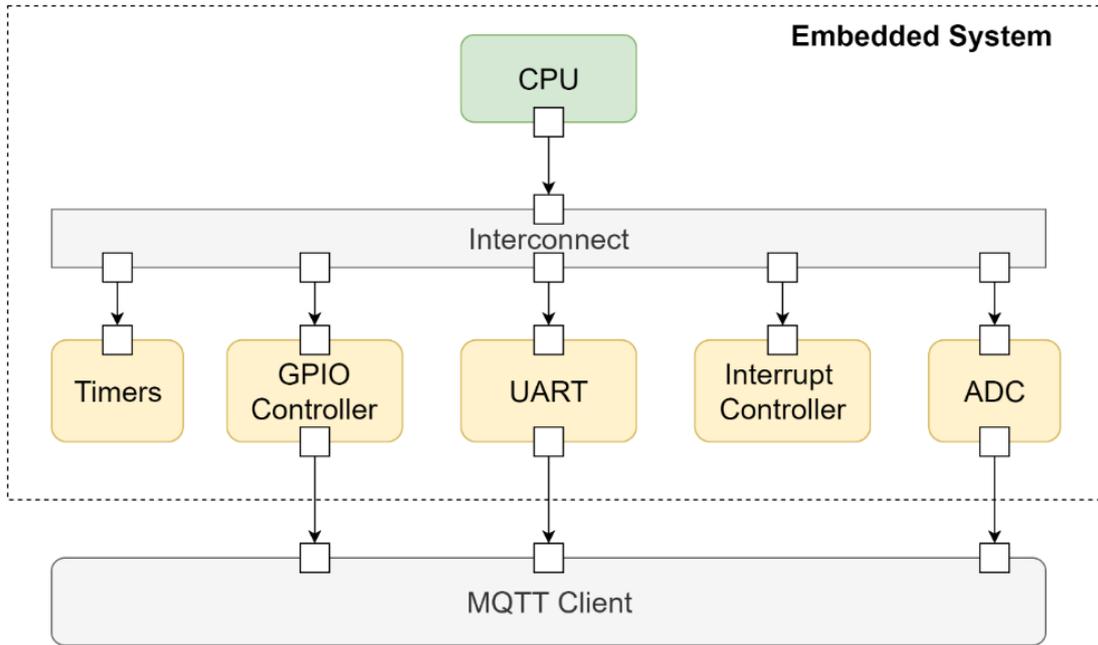


Figure 2: Programmable model of embedded system

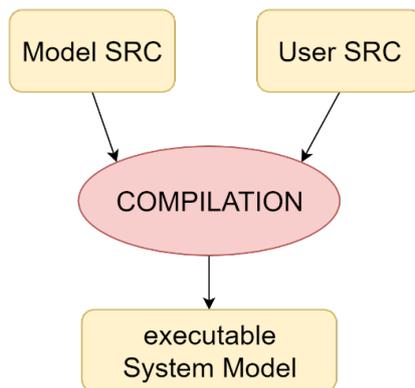


Figure 3: Getting executable model file

Table 2

Modes for communication via MQTT broker

MQTT channel	System Model role	User Session role
user_id/device	publisher	subscriber
user_id/web	subscriber	publisher

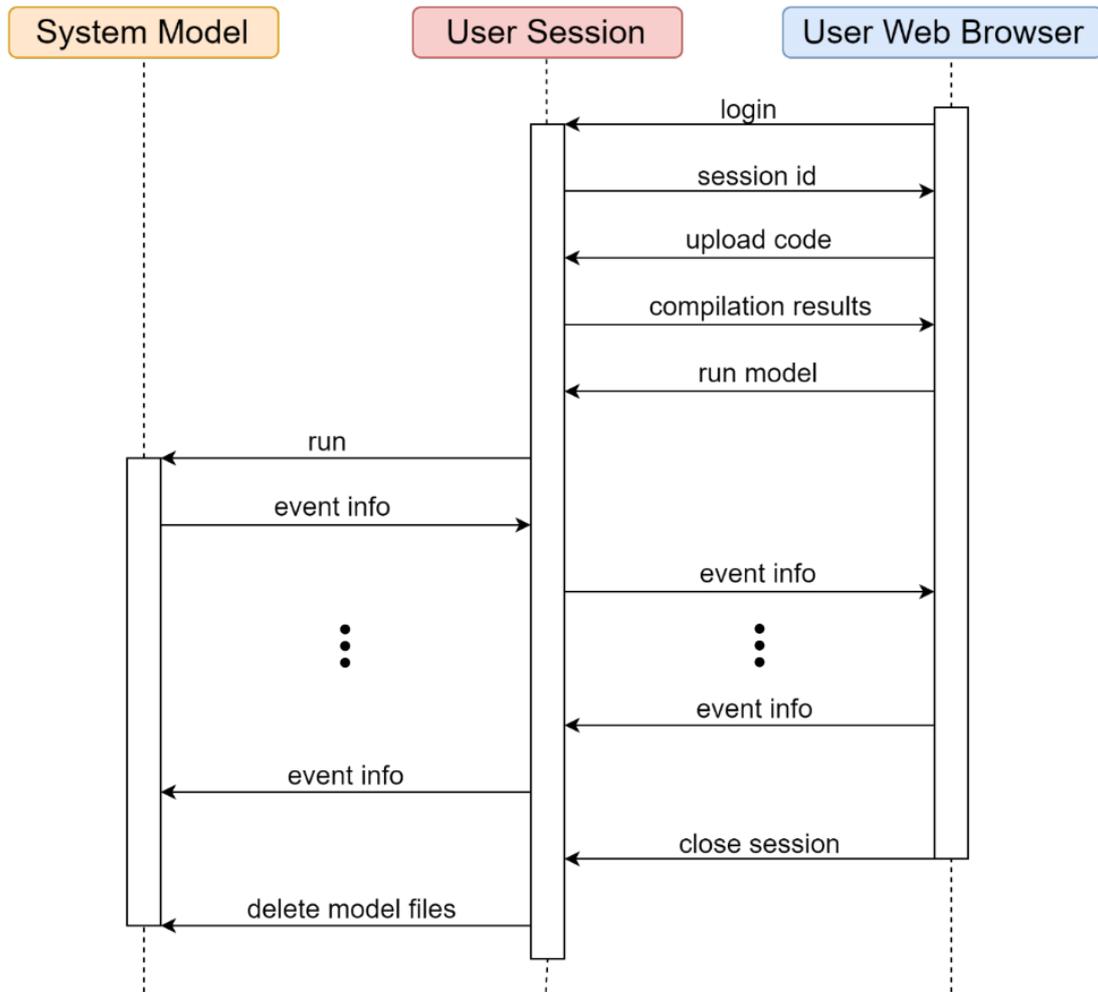


Figure 4: Interaction of virtual laboratory components

3.2. Evaluation

The webform of the developed simulation environment of SDK1.1M development board is presented in figure 5. The user can connect to the system remotely and upload the zip archive with sources. These sources can be used in the developed system and also in real STM32F4 series microcontrollers with minor modifications. After uploading the archive with source codes the user should compile the sources and run them on the board. During the program execution, the user can interact with the model using buttons in real-time. The model time in SystemC hardware models is equal to real-time.

The system has been installed on the server with Intel Core i5-4200U CPU with 2.3 GHz. This CPU consist of 4 logic cores and Tornado Web Server can be scaled on it. The Tornado Webserver can be scalable on all available CPU logic cores. The dependence of the user project

Virtual Environment

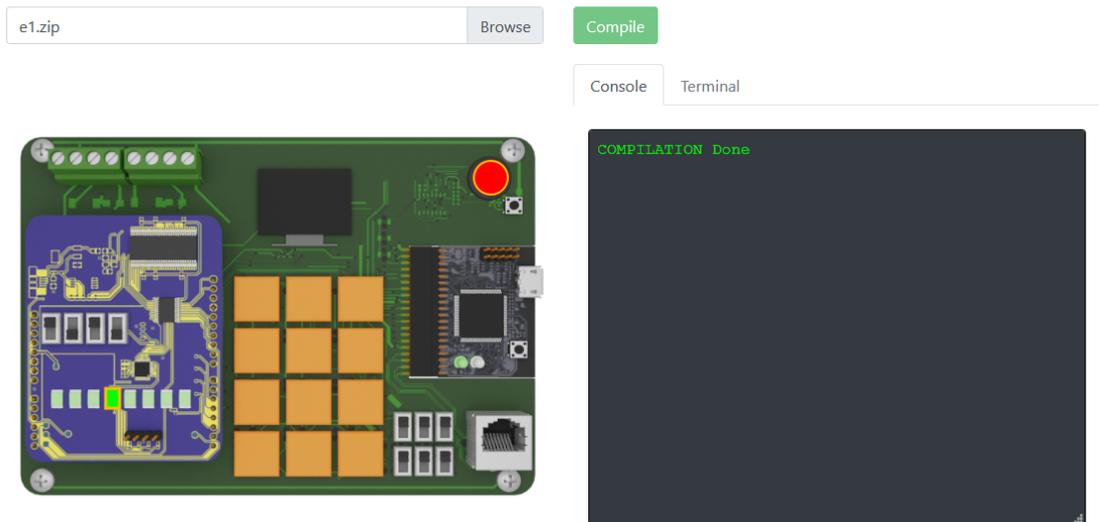


Figure 5: Virtual environment

compilation time on the number of parallel users is presented in figure 6.

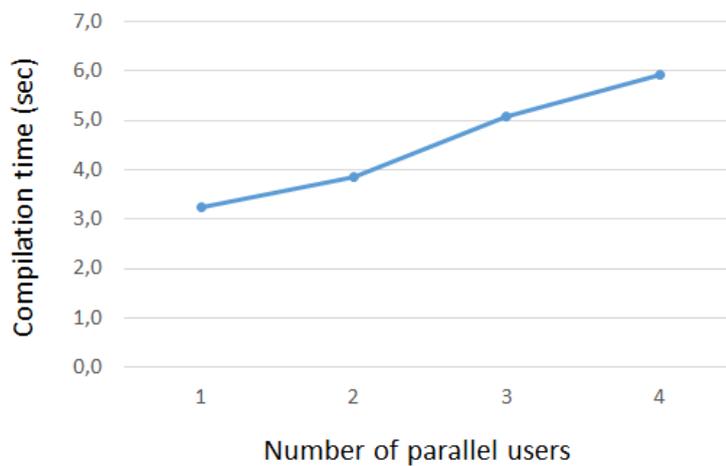


Figure 6: Dependence of compilation time on parallel users

There is a drawback of this solution that one user occupies one logic core of CPU. The overcoming of this drawback is the aim of further research.

The developed system is using in the Embedded Systems course on Open EDU platform³ based on Open edX.

³<https://openedu.ru/course/ITMOUniversity/EMBSYS/>

4. Results

In this article, we propose a scalable architecture for simulating microcontrollers with remote access. It can be scaled on logic cores of one server or several cores of several servers. The scalability is provided by using Tornado Web Server. The hardware model of a microcontroller is implemented as a loosely timed TLM model using SystemC library. SystemC provides a possibility to implement the model that can interact with a user in real-time.

Using the proposed architecture, the simulation environment of SDK1.1M development board based on STM32F4 microcontroller was developed and integrated into the Embedded System education course on the Open EDU platform.

The current drawback of the proposed system is the high processor load. It can be decreased by finding ways of effective data exchange methods between the server and the client sides. It is an issue for further research.

References

- [1] H. Zhou, Z. Xu, Z. Lin, R. Wang, J. Dong, Y. Huang, Simulation design of inverter in solar photovoltaic system based on mcu, Wuhan, 2009, pp. 1–4. doi:10.1109/SOPO.2009.5230265.
- [2] C. Bing-jie, Y. Li-chao, The construction of virtual lab base on proteus and keil, 3, 2009. doi:10.1145/1219092.1219093.
- [3] R. Grepl, Real-time control prototyping in matlab/simulink: Review of tools for research and education in mechatronics, IEEE, 2011.
- [4] Y.-H. L. Huang, Chih-Han, H.-L. Tsai, Design for microcontroller-based photovoltaic monitoring system using matlab/simulink, IEEE, 2016.
- [5] F. Bellard, Qemu, a fast and portable dynamic translator, volume 41, 2005.
- [6] D. Bartholomew, Qemu: a multihost, multitarget emulator, 145, 2006, pp. 68 – 71.
- [7] B. N. Mohapatra, R. K. Mohapatra, J. Joshi, S. Zagade, Easy performance based learning of arduino and sensors through tinkercad, volume 8, 2020, pp. 73 – 76.
- [8] B. N. Mohapatra, R. K. Mohapatra, J. Joshi, S. Zagade, Smart performance of virtual simulation experiments through arduino tinkercad circuits, volume 4, New York, NY, 2020, pp. 157 – 160.