An ARM Cortex M0 Port for Arduino with Infineon XMC1100
Sachin M S\textsuperscript{1} Dr. Shreenivas T H\textsuperscript{2}
\textsuperscript{1}M. Tech Student \textsuperscript{2}Professor
\textsuperscript{1,2}Department of Computer Science and Engineering
\textsuperscript{1,2}NIE Mysuru

Abstract— Microcontrollers are used in many fields such as infotainment systems, automobiles, security and in various other embedded system applications. With huge projects, code complexity becomes extremely high. Integrated Development Environment (IDE) tools make developers productive by helping them code and build projects with ease. ARDUINO IDE, being one of the exceptional examples. Arduino is a tool for making computers that can sense and control more of the physical world. The Arduino is implemented based on the Processing multimedia programming environment. Arduino is used for coding 8 bit microcontrollers, but the scope of this project extends the current 8 bit users an opportunity to explore the industrial grade 32 bit microcontrollers, without having to compromise in cost and efficiency but with greater ease of use and sophistication.

Key words: Arduino, IDE, ARM Cortex M0, Porting

I. INTRODUCTION
Increasing complexity and demand for computing power of embedded control applications requires microcontrollers to have a significant CPU performance, integrated peripheral functionality and rapid development environment enabling short time-to-market, without compromising cost efficiency. The XMC1100 series belongs to the XMC1000 Family of industrial microcontrollers based on the ARM Cortex-M0 processor core. The XMC1100 series devices are designed for general purpose applications.
Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. The Arduino integrated development environment (IDE) is a cross-platform application written in Java, and derives from the IDE for the Processing programming language and the Wiring projects. It is designed to introduce programming to artists and other newcomers unfamiliar with software development. It includes a code editor with features such as syntax highlighting, brace matching, and automatic indentation, and is also capable of compiling and uploading programs to the board with a single click. A program or code written for Arduino is called a sketch. Figure 1 shows the software architecture of the Arduino IDE.

Fig. 1: Arduino IDE Architecture

II. EXISTING SYSTEM TO PROGRAMME CORTEX M0 MICROCONTROLLER
Currently, the process of programming a XMC1100 (a Cortex M0 clone) Microcontroller is through DAVETM (Digital Application Virtual Engineer) tool developed and distributed by INFINEON Technologies. The DAVETM is platform dependent thus limiting the area of coding.
The Arduino IDE can only be used to programme ATMEL chips (Recently support has been extended for SAM Microcontrollers), there is no support for xmc family of microcontrollers. Arduino is open-source and cross-platform application provides support in major platforms like WINDOWS, UNIX and MACINTOSH.
Process of Coding a Cortex M0 device using DAVETM is as shown in figure 2.
Application developers write code in c files using peripheral specific applications provided with the DAVETM. Which is then compiled using ARM- GCC toolchain (GNU toolchain for ARM Cortex-M & Cortex-R processors) to generate binaries. The binaries are then flashed on to the memory of the microcontroller using flash loader programmes, such as jlink from segger.

Fig. 2: Programming Cortex M0 using DAVETM
III. PROPOSED SYSTEM TO PROGRAMME CORTEX M0 MICROCONTROLLERS

Developing a port for Arduino requires, making the Cortex M0 clone to be able to programme using wiring language. So there is a need for a software emulation layer called HAL (Hardware Abstraction Layer). To realise the porting, platform generic applications are to be developed, such as flash loader and bootloaders. It is as shown in figure 3.

IV. SCOPE OF WORK

The current methodology for programming a Cortex M0 clone using DAVE™ (as shown in figure 3) has some limitations, like DAVE being windows based application which makes it on-platform specific application. But the scope of the current on-going work empowers application developer to programme the microcontroller in wider platforms and with much ease of use (as shown in figure 4). Arduino being an open-source application, one cannot use licenced software such as flash loaders like Jlink.

In order to eliminate this dependency, a platform independent, open-source application is developed and provided.

V. METHODOLOGY

A. Eliminating DAVE Dependency:

Precisely, to make porting possible, certain device specific files (Generated by DAVE resolver) and Peripheral application dependency is to be removed. Isolation of the LLD (Low Level Drivers) is an important aspect in order to develop HAL.
B. Introducing Arduino:

Arduino IDE matches the exact requirement of achieving cross-platform compatibility along with ease of use, which reduces the time consumption for SDLC (Software Development Life Cycle). Arduino IDE already being an established platform for Microcontroller programming with huge support for accessories (Shields & libraries) provides a perfect housing for the development of microcontroller based systems. In order to achieve platform independency certain level of abstraction and development is required which will be explained in the later part of the paper.

C. Implementation:

This section explains how exactly portability is achieved and integration is made possible within wiring environment. It involves following step.

1) Development of HAL:

Hardware abstraction layer (HAL) is a layer of programming that allows API's (Application Specific Interfaces) to interact with a hardware device at a general or abstract level rather than at a detailed hardware level. In order to retain compatibility with Arduino specific wiring language API's, a wide range of abstraction is required on LLD’s (Low Level Drivers).

Vendor independent hardware abstraction is provided by ARM, which is known as CMSIS (ARM® Cortex™ Microcontroller Software Interface Standard). This can be used to provide abstraction at scratch level and to retain compatibility with all Cortex-M family of microcontrollers. The peripheral specific API’s are implemented in accordance with the application specific requirements and abstraction is developed based on wiring library API’s implementation.

D. Replacing Flashloader and Bootloaders:

A bootloader is a program that runs in the microcontroller to be programmed. It receives new program information externally via some communication means and writes that information to the program memory of the processor. Bootloader has to be written to the flash memory just once using a conventional programmer. The bootloader is programmed such that when “bootloader start condition” is satisfied it receives data via a Predetermined interface (UART) and writes these into the program memory at predetermined Locations.

DAVE uses ASC-BSL (Asynchronous Serial Interface Bootstrap Loader). This will be replaced with a platform independent Bootstrap program to provide communication to the MCU (Microcontroller unit).

The flash loader is a small piece of code that allows programming of the flash without the need for a debugger interface. The new Flashloader being platform independent and provides command line interface.

VI. RESULTS AND COMPARISON

This section shall spot the light on the work carried to the existing process of conventional programming using Dave to that of the making computers using Arduino.

The results and comparison can be made briefly thinking the following aspects.

A. Minimal Coding Maximum Gain:

Just to give an example of the complexity in coding using Dave, to write a blinky on an xmc device shall take a code length of 15 – 20 lines. But on the other hand the same can be achieved by 4 line of code using Arduino ide.

By this minimalistic comparison we can determine how much of ease is achieved through this work in programming the devices.

B. Projects Creation

Arduino provides so many shields and online forum for exiting projects such as robots, quadcopters etc., since xmc being a real powerful industrial grade microcontroller, the efficiency of the created project will be more and efficient.

C. Time Consumption

Since with minimal coding we can achieve maximum gain the time that is consumed in creating a system would be much lesser that of the conventional coding with Dave.

D. Resource Availability

Arduino is being open source, many individuals can develop their own shields and standards for project creation, since there are numerous such enthusiastic users, and the resource availability along with online help is much greater than that of Dave.

VII. CONCLUSION

This porting shall provide ease of use for the developers to create interactive systems. The time consumption and code complexity decreases dramatically and resulting in better performance.

REFERENCES
