BeagleBoard/GSoC/2021 Proposal/OmkarBhilare

< BeagleBoard | GSoC

Contents

Proposal for Beaglewire Software

About *Student*: Omkar Bhilare (https://elinux.org/User:OmkarBhilare)

Mentors: Michael Welling (https://elinux.org/User:M_w)

Proposal:BeagleWire Software (https://elinux.org/BeagleBoard/GSoC/2021_Proposal/OmkarBhilare)

Code: BeagleWire Code (https://github.com/ombhilare999/BeagleWire) *Wiki*: BeagleWire Software (https://ombhilare999.github.io/GSoC-2021/) *GSoC*: BeagleWire GSoC Project (https://summerofcode.withgoogle.com/projects/#4845032915337216)

Status

This project is Selected for GSoC 2021.
Logs can be found here: https://ombhilare999.github.io/GSoC-2021/

Proposal

- Completed All the requirements listed on the ideas page.
- The PR request for cross-compilation task: #154 (https://github.com/jadonk/gsoc-application/pull/154).

About you

IRC: Omkar Bhilare [@ombhilare99:matrix.org]
Github: ombhilare999 (https://github.com/ombhilare999)
School: Veermata Jijabai Technological Institute (http://vjti.ac.in/)
Country: India
Primary language : English, Hindi, Marathi
Typical work hours: 10AM-8PM Indian Standard Time
Previous GSoC participation: This is my first time applying for GSoC, I'm an Electronics enthusiast and have a great interest in fields like FPGA, Digital VLSI, Computer Architecture. I have experienced with Intel's Quartus, Xilinx's vivado, and opensource toolchains for ice40. I have also done many projects related to Cyclone2, Upduino3.0(ICE40), Sipeed Tang Primer FPGA.

About your project

- *Project name*: Beaglewire Software

Description

Introduction

The BeagleWire is an FPGA development platform that has been designed for use with BeagleBone boards. BeagleWire is a cape on which there is an FPGA device (Lattice iCE40HX). The software support for BeagleWire is still in the development phase. In this project, I'm developing and testing the existing software support of Beaglewire.

The known primary issue in Beaglewire is the interface between 32MB SDRAM and ICE40HX4K. For this Solution, I'm going to try options like LiteDRAM(a small footprint and configurable DRAM core) or Advanced SDRAM controller provided by LATTICE.

Why LiteDRAM (https://github.com/enjoy-digital/litadram) for SDRAM Control

The Core produced by LiteDRAM is

1. Fully pipelined, high performance.
2. Configurable commands depth on bankmachines.
3. Auto-Precharge.
4. Periodic refresh/ZQ short calibration (up to 8 postponed refreshes).

(;LiteDRAM is already used in commercial and open-source designs)

Testing and Improvement of Subsystem Like I2C, SPI, PWM, UART

In this project I'm going to test all the subsystems like I2C, SPI, PWM, UART in Hardware and the primary goal will be to debug the issues related to it and fix them accordingly. Along with the driver code ready to use solutions will be added in software support.

- I2C: There are two grove Connectors on BeagleWire for I2C, I will test the current Verilog Code of I2C in hardware, and will found out if further improvements can be done or not.
- UART: The existing code of Uart will be tested in hardware, and few examples also will be added for UART.
- PWM: New Example will be created for testing of PWM. Servo Code will serve the purpose of testing the PWM drivers.

Others

- There are current Issues open on BeagleWire Repository (https://github.com/pmezydlo/BeagleWire/issues), These will be solved during this project.
- More PMODs will be interfaced with the BeagleWire.
- Increase the Documentation and also add getting started guide for BeagleWire.

Details of Implementation

- First of all, I need to load a pre-prepared BBB image to beagle with existing all required drivers and scripts to start BeagleWire development.

The steps give on Quick Start Guide (https://elinux.org/BeagleBoard/BeagleWire#Quick_Start_Guide):

- To generate the bitstream I'm going to first try the existing fpga-load (https://github.com/mwelling/BeagleWire/blob/master/load_fw/fpga-load.c) script which leverages the FPGA manager, which is the core that exports a set of functions for programming an FPGA with an Image.

LiteDRAM

1. LiteDRAM provides a small footprint and configurable DRAM core. The configuration of LiteDRAM can be changed using yml input. The examples (https://github.com/enjoy-digital/litedram/tree/master/examples) given the repository as follows:

```
#
# This file is part of LiteDRAM.
#
# Copyright (c) 2018-2019 Florent Kermarrec <florent@enjoy-digital.fr>
# SPDX-License-Identifier: BSD-2-Clause

{
    # General ------------------------------------------------------------------
    "cpu":        "vexriscv",  # Type of CPU used for init/calib (vexriscv, lm32)
    "speedgrade": -1,          # FPGA speedgrade
    "memtype":    "DDR2",      # DRAM type
.
.
.
```

2. This configuration can be changed for this application that is memory type to SDRAM and CPU None:

```
cpu=None
```

Testing of Subsystems

1. For this I need to synthesize the Verilog code first using IceStorm Toolchains, I will be first referring to the steps mentioned in the BeagleWire Documentation:
Synthesizing Verilog Code using Icestorm Toolchains (https://elinux.org/BeagleBoard/BeagleWire#Synthesizing_Verilog_code_using_IceStorm_toolchain)
2. Using this I can test the driver codes in hardware and observe the waveforms on DSO if needed.

PMOD Support

1. The BeagleWire has the PMOD connectors reversed from the "Standard" template.
The standard template is as follows:

```
3.3V | GND | 3 | 2 | 1 | 0
3.3V | GND | 7 | 6 | 5 | 4
```

The pinout for BeagleWire PMOD:

```
4 | 5 | 6 | 7 | GND | 3.3V
0 | 1 | 2 | 3 | GND | 3.3V
```

2. So to use PMODs with BeagleWire One needs to connect the PMOD upside down or use some sort of Breakout Board. I found this Breakout made out especially for BeagleWire: BeagleWire to PMOD (https://github.com/CapableRobot/PMOD-Adapters#cr7csz-beaglewire-to-pmod)
3. Using this Breakout board I will be Interfacing New PMODs to BeagleWire, changes will be done in the current pcf file for GPIO position if found wrong.

Hardware Needed

1. BeagleWire
2. BeagleBone Black / BeagleBone Black Wireless
3. Various PMODs
4. PMOD Breakout Board

Timeline

Date	Status	Details
13 April :: 17 May	Application Review Period	▪ Go Through BeagleWire Docs (https://elinux.org/BeagleBoard/BeagleWire) ▪ Go Through Current BeagleWire Issues (https://github.com/pmezydlo/BeagleWire/issues)
15 May :: 7 June	Community Bonding	▪ Discuss with a mentor about the doubts related to implementation. ▪ Setting Up BeagleBone with the Image used by Michael Welling (https://elinux.org/User:M_w) for testing BeagleWire Boards. ▪ Documenting the Process for getting started guide.
June 14	Milestone #1	▪ ~~Introductory YouTube video~~ ▪ ~~Make Sure Everything setup correctly on BeagleWire and able to generate bitstream.~~ ▪ ~~Running *blink_leds* (https://github.com/pmezydlo/BeagleWire/tree/master/examples/blink_leds) code from BeagleWire Repo.~~
June 21	Milestone #2	▪ Generate the LiteDRAM Core with the Memory Type: SDRAM and CPU: NONE ▪ ~~Understand the Current SDRAM Controller (https://github.com/pmezydlo/BeagleWire/blob/master/components/sdram_controller.v).~~ ▪ Interface the LiteDRAM Core with other firmware
June 28	Milestone #3	▪ Testing the SDRAM in Hardware. ▪ Understand and try to solve the Issues related to SDRAM on the BeagleWire repo: 1. sdram read results are occasionally wrong #7 (https://github.com/pmezydlo/BeagleWire/issues/7) 2. sdram rd_busy should be rd_ready? #8 (https://github.com/pmezydlo/BeagleWire/issues/8) 3. sdram can miss command transitions #10 (https://github.com/pmezydlo/BeagleWire/issues/10) ▪ In case of any issues arises in LiteDRAM Core also check the backup of Advanced SDRAM Controller on Lattice Sites.
July 5	Milestone #4	▪ Test and Improve Following Subsystems: 1. *SPI* (https://github.com/pmezydlo/BeagleWire/tree/master/examples/spi) 2. *UART* (https://github.com/pmezydlo/BeagleWire/tree/master/examples/uart)
July 12 - 16, 2021	Milestone #5(First Evaluation)	▪ Demonstrating the use of SDRAM on BeagleWire, SPI, and UART

		▪ After the review from the mentor, finalizing and Documenting Everything done till now
July 23	Milestone #6	▪ Test and Improve Following Subsystems: 1. *PWM* (https://github.com/pmezydlo/BeagleWire/tree/master/examples/pwm) (Write Servo Code using PWM) 2. *I2C* (https://github.com/pmezydlo/BeagleWire/tree/master/examples/i2c)
July 30	Milestone #7	▪ Test the other Examples provided in the BeagleWire Repo: 1. stepper motor (https://github.com/pmezydlo/BeagleWire/tree/master/examples/stepper_motor) 2. lcd_game (https://github.com/pmezydlo/BeagleWire/tree/master/examples/lcd_game) ▪ Add New Examples for the BeagleWire using updated subsystem codes.
August 6	Milestone #8	▪ Interface Various PMODs with BeagleWire. ▪ Use the Breakout Board with Beaglewire for Hardware testing of PMODs.
August 16	Milestone #9	▪ Solve all the issues opened on the BeagleWire Repo: Issues (https://github.com/pmezydlo/BeagleWire/issues) ▪ Document Everything Until Now. ▪ Add final About page.
August 21	Milestone #10	▪ Submit final work product and final mentor evaluation ▪ Completion YouTube video
August 31	Milestone #11	▪ Completion of GSoC

Experience and approach

- I'm well experienced with Verilog and C. I have actual experience with working with FPGAs. I have worked with Intel's Cyclone2, Sipeed's Tang primer, and Upduino3.0(ICE40) FPGA Boards.
- I have done several projects related to these boards with Verilog some of them are as follows:

1. riscv-core in verilog (https://github.com/ombhilare999/riscv-core)
2. VGA Interface With Tang Primer FPGA (https://github.com/ombhilare999/vga-interface-with-TANG-PRIMER-FPGA)
3. Seven Segment Interface with Tang Primer FPGA (https://github.com/ombhilare999/Seven-Segment-with-Tang-Primer-FPGA)

More projects done by me can be found on my github (https://github.com/ombhilare999)

- I have also designed various double-sided boards using Autodesk Eagle and Kicad.
- Esp32 Development Board designed for Embedded and Robotics Application: Design (https://github.com/SRA-VJTI/sra-board-hardware-design), This is the last Board I had designed, this shows that I have a very good understanding of reading schematics which was one of the requirements of the project

Contingency

if I get stuck on my project and my mentor isn't around, I will use the following resources:

1. Getting Started Guide for BeagleBone by derek molloy: http://derekmolloy.ie/beaglebone (http://derekmolloy.ie/beaglebone)

- Current Github Repos:

1. https://github.com/pmezydlo/BeagleWire (https://github.com/pmezydlo/BeagleWire)
2. https://github.com/mwelling/BeagleWire (https://github.com/mwelling/BeagleWire)
3. https://github.com/osresearch/BeagleWire (https://github.com/osresearch/BeagleWire)

- Documentation on the pmezydlo repo of BeagleWire: Documentation (https://github.com/pmezydlo/BeagleWire/tree/master/documentation)
- BeagleWire Page (https://elinux.org/BeagleBoard/BeagleWire)

Benefit

The completed project will provide the BeagleBoard.org community with easy to implement and powerful tools for the realization of projects based on Programmable Logic Device(FPGA), which will surely increase the number of applications based on it. The developed software will be easy and, at the same time, efficient tool for communication with FPGA. At this point, FPGA will be able to meet the requirements of even more advanced applications. The BeagleWire creates a powerful and versatile digital cape for users to create their imaginative digital designs.
