Применение открытых IP ядер для построения СнК с Linux

Микроэлектроника 2023

Монахов А.М., Михайлов Д.В., Лукьянченко Г.А., Елизаров С.Г.

ООО «Мальт систем»

MALT system

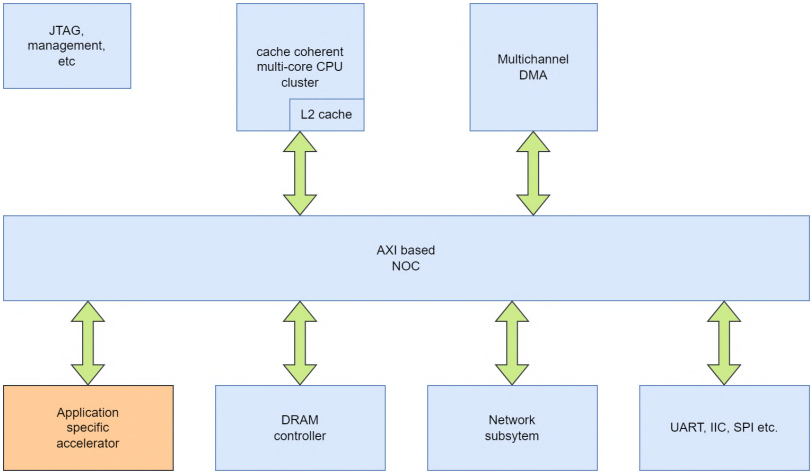
Содержание

РОССИЙСКИЙ ФОРУМ МИКРОЭЛЕКТРОНИКА 2023

Введение

- Типовой задачей для многих дизайн-центров является разработка СнК с процессором общего назначения и специализированной частью (ускорителями, аналоговыми блоками и т.д.);
- Для многих современных встраиваемых задач стандартом стало использование Linux, что позволяет значительно упростить разработку ПО;
- Стандартной практикой для решения подобных задач является лицензирование процессорных ядер с архитектурой ARM и вспомогательных IP для них;
- Крайне важным параметром подобных проектов является скорость получения макета системы для отладки ПО;
- На сегодняшний день появилась альтернатива лицензированию проприетарных ядер в виде использования свободно распространяемых IP-блоков.

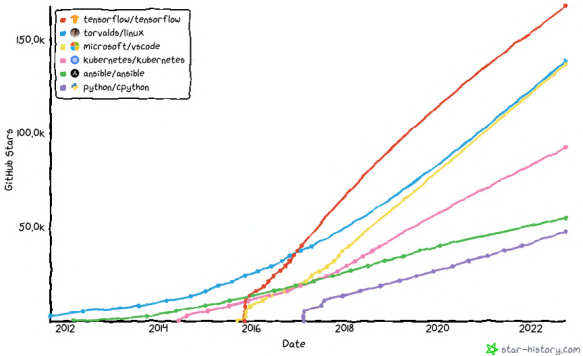
Типовая задача по проектированию SoC

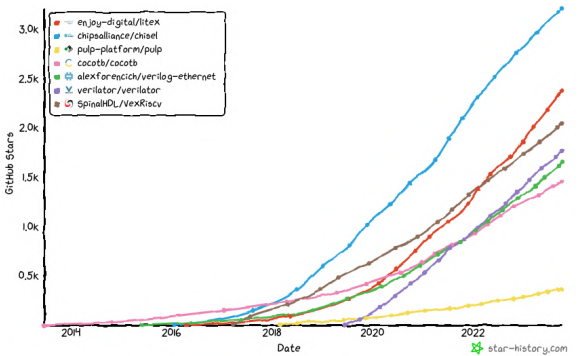


Открытые библиотеки IP ядер

Рост популярности открытых IP-ядер

РОССИЙСКИЙ ФОРУМ МИКРОЭЛЕКТРОНИКА 2023

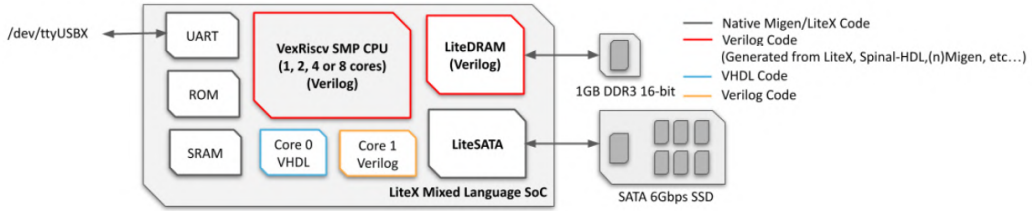




LiteX

LiteX - инструмент для построения SoC, на HDL migen базирующемся на Python. Нацелен в первую очередь на FPGA.

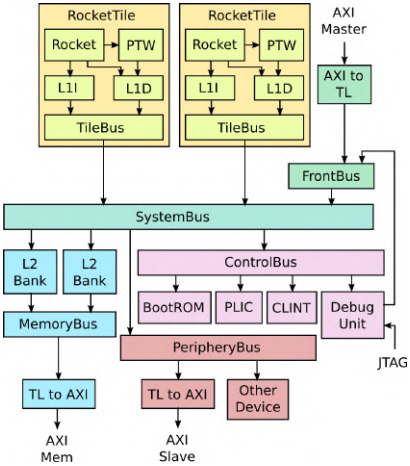
- Простота миграции с платформы на платформу;
- Богатая библиотека шинных блоков и периферийных контроллеров;
- Быстрое прототипирование;
- Возможность интеграции блоков на других HDL.



Chipyard

Chipyard - инструмент для построения SoC, на HDL Chisel базирующемся на Scala.

- Удобство миграции с ПЛИС на кремний;
- Широкий набор шинных и периферийных блоков;
- Поддерживает подключение высокопроизводительных процессорных ядер;
- Chisel используется для коммерческой разработки в SiFive.



PULP

Набор компонентов шинной AXI-подсистемы.

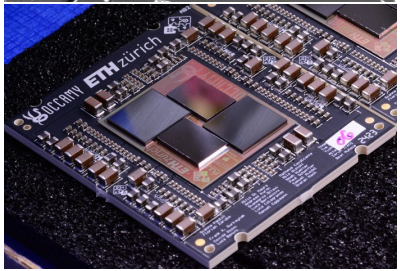
- Написаны на поддерживаемом САПР подмножестве System Verilog;
- Содержат такие компоненты как коммутаторы, буферы и конвертеры AXI;
- Удобно конфигурируются;
- Удобно интегрируются;
- Silicon-proven.

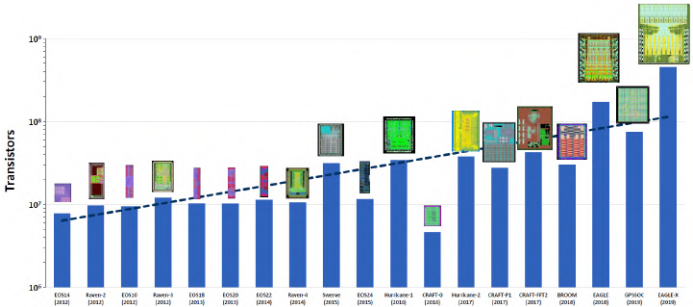
Verilog ethernet components

Набор компонентов Etheret подсистемы от автора opensource NIC Corundum.

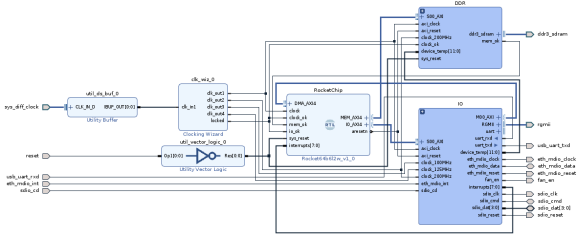
- Блоки MAC/PCS для 10 и 25GE;
- Верификационое окружение для cocotb с агентами AXI, AXIS, AXIL, различными MII;
- Демодизайны на множество популярных отладочных плат как на Xilinx так и на Intel.

Кремниевые реализации





Vivado Block Design vs LiteX




```
class BaseSoC(SoCCore):
  def __init__(self,sys_clk_freq=100e6,**kwargs):
    platform = digilent_genesys2.Platform()
    self.crg = _CRG(platform, sys_clk_freq)
    # SoCCore
    SoCCore.__init__(self, platform, sys_clk_freq,
      ident="LiteX on Genesys2", **kwargs)
    # DDR3 SDRAM
    if not self.integrated_main_ram_size:
      self.ddrphy = s7ddrphy.K7DDRPHY(platform.request("ram"),
      memtype      = "DDR3",
      nphases      = 4,
      sys_clk_freq = sys_clk_freq)
      self.add_sdram("sdram",
      phy           = self.ddrphy,
      module        = MT41J256M16(sys_clk_freq, "1:4"),
      l2_cache_size = kwargs.get("l2_size", 8192)
    )
    # Ethernet
    if with_ethernet:
      self.ethphy = LiteEthPHYRGMII(
      clock_pads = self.platform.request("eth_clocks"),
      pads       = self.platform.request("eth"))
      self.add_ethernet(phy=self.ethphy)
```

Поддерживаемые open source генераторами платформы

РОССИЙСКИЙ ФОРУМ МИКРОЭЛЕКТРОНИКА 2023

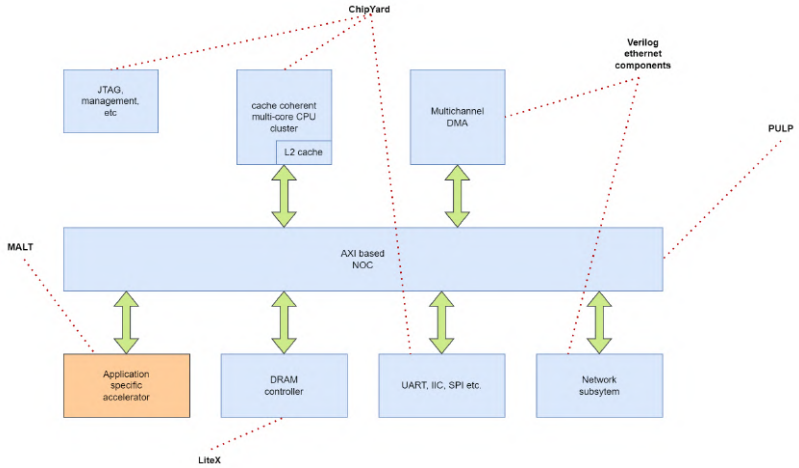
- Xilinx AC701
- Xilinx Alveo-U200
- Xilinx KCU105
- Xilinx SP605
- Xilinx VC707
- Digilent Arty
- Digilent Genesys 2
- Digilent Zybo Z7
- Alinx AXKU040
- ...

- Terasic DE0 Nano
- Terasic DE1
- Terasic DE2 115
- Terasic DE10 Nano
- Terasic DE10 Lite
- QMTech EP4CEX5
- Trenz MAX1000
- ...

- Efinix Triton dev kit
- Icebreaker (iCE40)
- Lattice Versa (ECP5)
- Colorlight i5 (ECP5)
- Sipeed Tang Nano 9k (Gowin)
- И еще десятки других

Что нам стоит SoC построить

Открытые IP в составе типового SoC

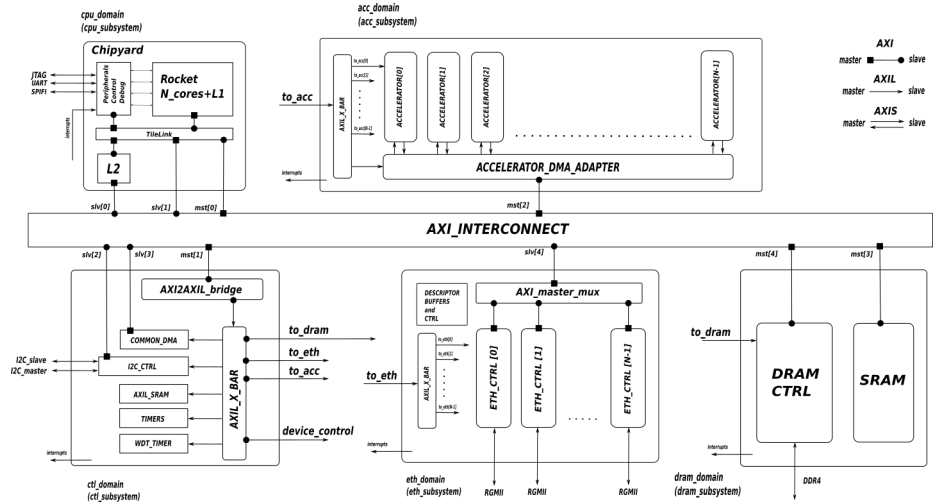


Синтез в СБИС

Все использованные нами компоненты подходят для синтеза в СБИС:

- Высокоуровневые генераторы (такие как Chisel) генерируют Verilog-2001;
- PULP написан на поддерживаемом подмножестве System Verilog;
- Макросы памяти и падов в Chisel удобно заменяются на ASIC-варианты;
- Процессорные ядра и шинная инфраструктура конвейеризированы и позволяют достичь частот 1 ГГц на 28 нм.

Структурная схема нашего дизайна



Софт

- Linux доминирует в embedded приложениях;
- Возможность использования mainline ядра Linux и дистрибутивов удобна для разработчиков;
- Использование mainline ядра Linux позволяет обновлять софт из основных репозиториев;
- В нашем случае адаптации потребовали только загрузчики (first-stage, Uboot) и эмулятор (QEMU);
- Адаптированный QEMU позволяет точно имитировать систему на всех этапах работы, включая загрузку и дает возможность подключать модули эмуляции специализированных ускорителей.

Верификация RTL

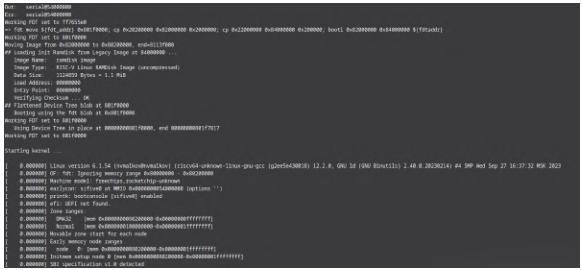
- Основной фреймворк: *Cocotb*
- VIP для шинной инфраструктуры: *Cocotb-AXI*
- Периферийные VIP: *Ethernet, I2C, SPI и др*.
- Собственные VIP для проверки: *TCP, спец. ускорители и др.*


```
async def simple_transmit (dut):
    my_env = TB(dut)
    await my_env.Run()

    server = my_env.connection_1.create_socket("SERVER",
      "base_IPv4", "172.16.2.211",1)
    client = my_env.connection_0.create_socket("CLIENT[0]",
      "base_IPv4", "198.168.1.3",666)

    client_conn = client.CreateConnection("172.16.2.211",1)
    server_conn = server.ListenConnection()
    await my_env.wait_some_time()
    await client_conn.TcpSendIntArray(
      [0xa1a2a3a4,0xb1b2b3b4,0xc1c2c3c4])

    server_conn.TcpShowAppBuffer()
    client_conn.TcpActiveClose()
    await my_env.wait_some_time()
#-------------------------------------------------------------
if coco.SIM_NAME:
    factory = TestFactory(simple_transmit)
    factory.generate_tests()
```

Верификация на ПЛИС

РОССИЙСКИЙ ФОРУМ МИКРОЭЛЕКТРОНИКА 2023

- Первый макет на ПЛИС с запущенным Linux был получен в первый месяц разработки;
- Макет позволил провести тесты периферии еще до того как были написаны модульные тесты RTL;
- Производительность CPU соответствует оценочной: 2.3 Coremark/МГц на ядро.
- Нагрузочные тесты на макете не выявили ни одной аппаратной проблемы в разрабатываемой SoC.



Заключение

Порог входа

- Излишне лаконичная документация;
- Отсутствие техподдержки;
- Необходимость иметь специалистов знакомых с высокоуровневыми HDL, такими как Chisel;
- Требуется тщательное тестирование при стыковке блоков из разных библиотек;
- Требуется адаптировать низкоуровневое ПО, например загрузчик ОС.

Выводы

- Библиотеки открытых IP пригодны для использования в промышленной разработке СБИС;
- С использованием этих библиотек возможно быстро получать прототип разрабатываемой СнК;
- Открытые верификационные библиотеки позволяют получать полноценное тестовое окружение и разрабатывать собственные VIP;
- Использование стандартного стека софта упрощает разработку и отладку целевого ПО, в том числе на ранних этапах проекта;
- В нашей компании накоплен опыт позволяющий быстро разрабатывать SoC с использованием открытых библиотек IP ядер. Такой SoC совместим с mainline Linux, а кастомизации требуют лишь драйвера для специализированных под задачу заказчика блоков.

Спасибо за внимание!