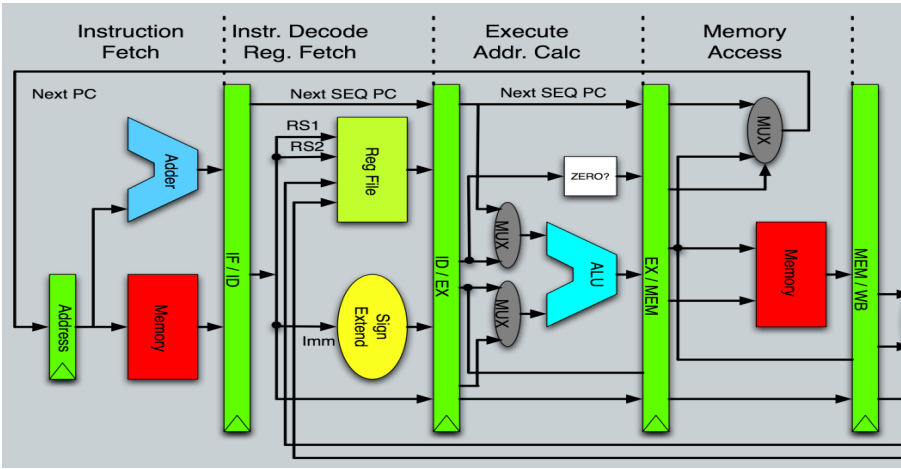
Mini-MIPS project
(Steffen Malkowsky)

- 32-bit RISC with a subset of MIPS instructions.
- Grading:
  - Grade 3: Fully verified pipelined Mini-MIPS.
  - Grade 4 : Xilinx Ethernet I/O
  - Grade 5 : Extra functionality (accelerator/extended instruction set
  - Alternative for Grade 4/5
    - Own ideas from you!
- Prerequisite course:
  - EITF20 Computer Architecture

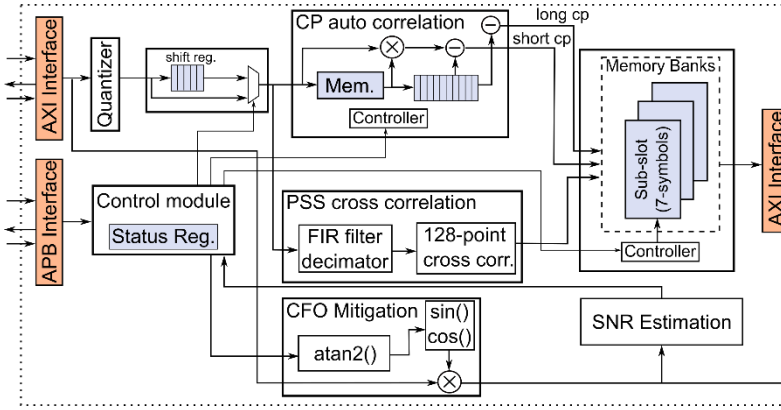


RISCV SoC: Hardware and Software Integration(Hemanth Prabhu)

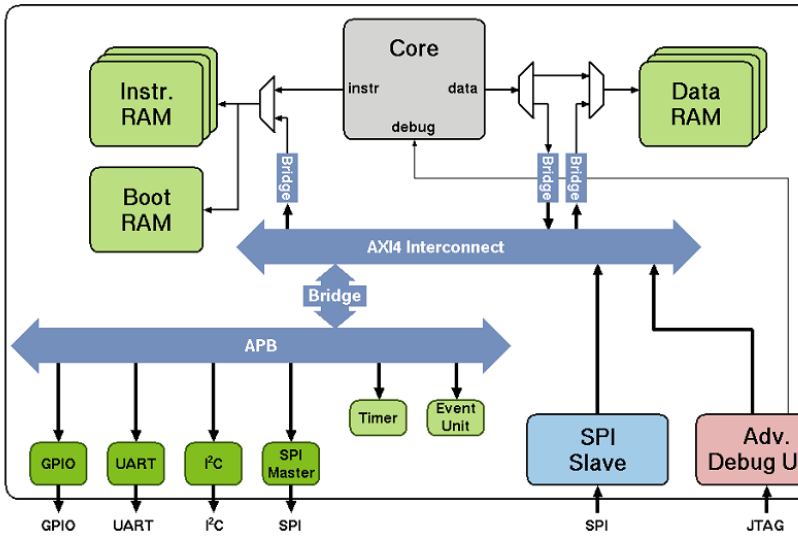
- Primary tasks for grade 5, is to integrate an existing synchronization module:
  - **Implement** AXI and APB interfaces for sync module (grade 3)
  - **Integrate** module into PULPino SoC (grade 4)
  - **Verify** that the processor core (RISCV) is able to access sync module
  - **Implement/Integrate** DMA (grade 5)

- Will require both hardware and software development skills! (select one group based on assignment report quality)

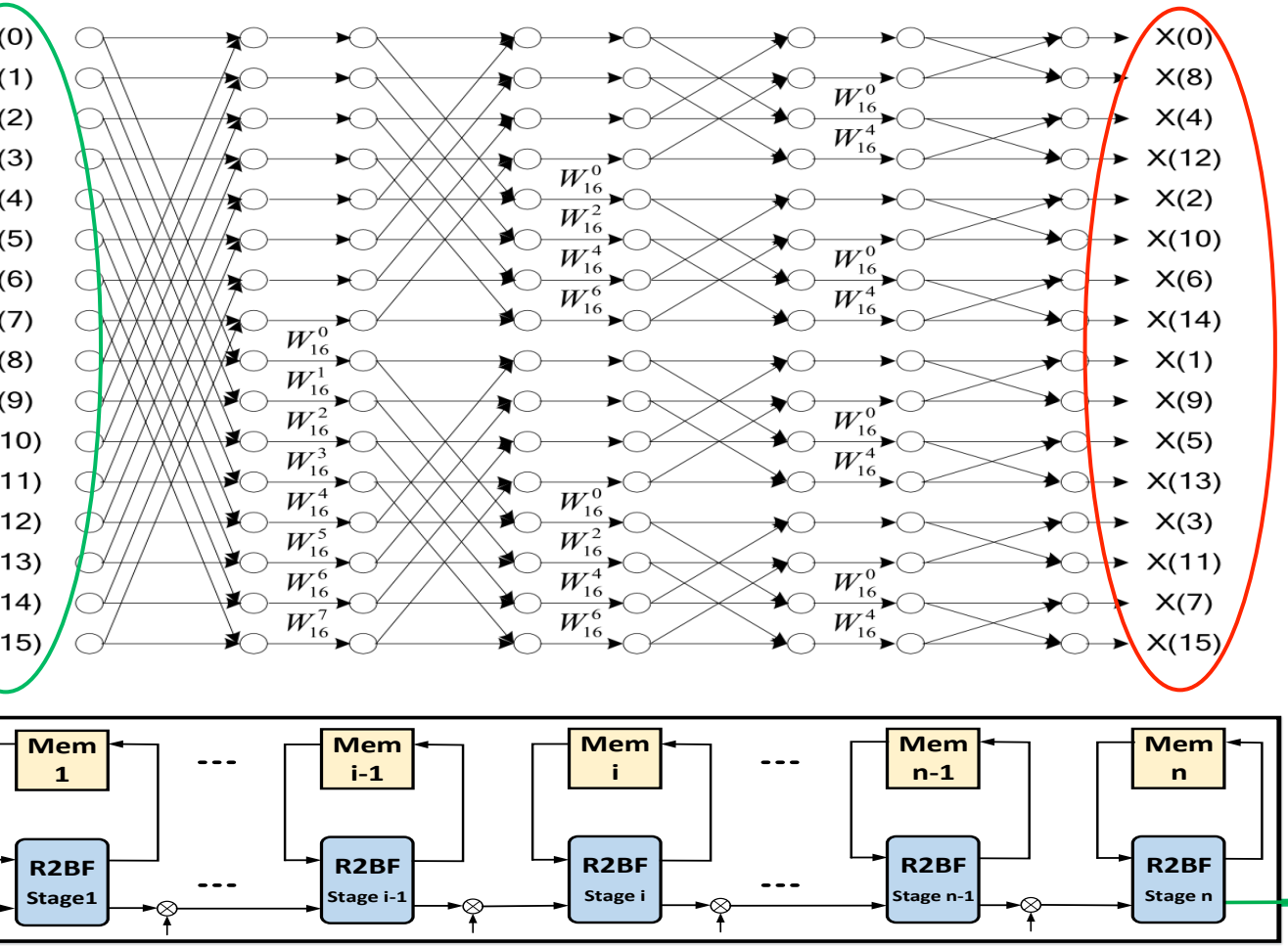
Synchronization module



PULPino SoC



Reordering Circuit for Pipelined FFT(Mojtaba Mahdavi)



Pipelined FFT

- Inputs are in **Normal order**:
  - X0, X1, X2, …, X15
- Outputs are in **Bit-Reversed order**:
  - X0, X8, X4, …, X15
- *Reordering Circuit is needed*

Reordering Circuit

Reordering Circuit

Example:
Input/output of reordering circuit for 16-point FFT

Input	Input Index	Output	Output Index
X0	0000	X0	0000
X8	1000	X1	0001
X4	0100	X2	0010
X12	1100	X3	0011
X2	0010	X4	0100
X10	1010	X5	0101
X6	0110	X6	0110
X14	1110	X7	0111
X1	0001	X8	1000
X9	1001	X9	1001
X5	0101	X10	1010
X13	1101	X11	1011
X3	0011	X12	1100
X11	1011	X13	1101
X7	0111	X14	1110
X15	1111	X15	1111

- There are different methods for reordering from the **Bit-Reversed Order** to **Normal Order**.
  - Implement an efficient reordering circuit for:
    - 2048-point FFT
    - One or two input per clock
    - FFT implementation is **NOT** required
    - *Just implementation of a reordering circuit is needed*

IC PROJECT - CRYPTOGRAPHY

Implementation of a stream cipher and related attack

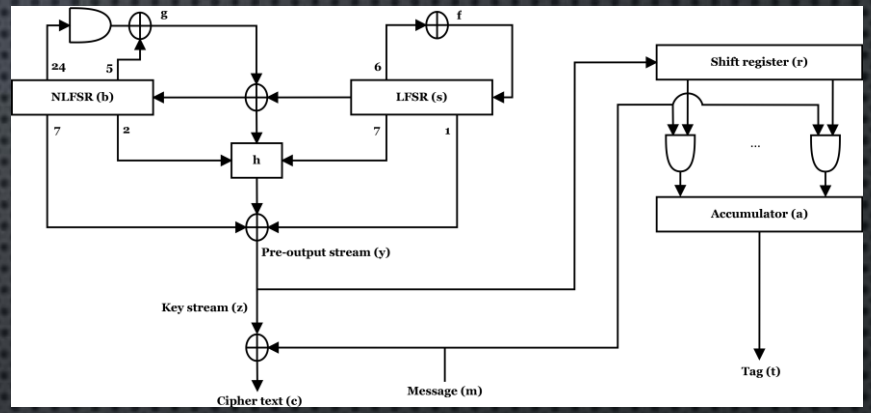
Jonathan Sönnerup,

jonathan.sonnerup@eit.lth.se

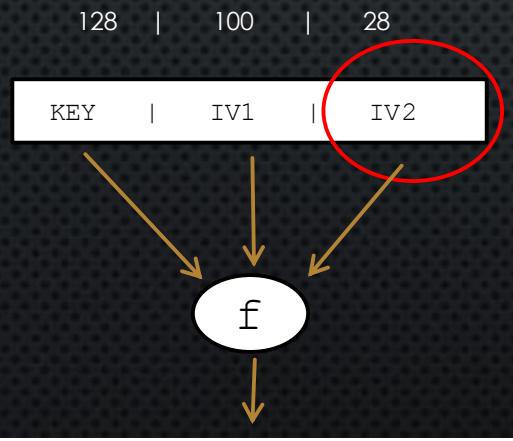
GOAL

- The task is to implement a part of a generic state-of-the-art attack against cryptographic functions, focusing on the stream cipher Grain 128a:
  1. Implement Grain 128a with authentication (3).
  2. Parallelize Grain up to a factor 32 efficiently (4).
  3. Implement a part of an attack against Grain, known as the Maximum Degree Monomial test (5).

Grain 128a w/ auth



MDM test



x0	x1	y
0	0	1
0	1	0
1	0	1
1	1	1

Sum(y) = 0000000000000010110...
(MDM signature)

$$z_0 = \alpha_{0100} k_1 + \alpha_{1000} k_2 + ... + \alpha_{0001} IV_1 + \alpha_{0010} IV_2 + ... + \alpha_{0101} k_1 IV_1 + ... + \alpha_{1111} k_1 k_2 IV_1 IV_2$$

Parallelized Grain