Nios II Flash Accelerator Using Max10

2015.06.30

AN740 Subscribe Send Feedback

Introduction

As part of the initiative to improve Nios II/f "fast" core performance in real time applications, the Nios II flash accelerator is introduced. The accelerator is optimized to fetch instructions from flash memory and cache them in registers for fast instruction access.

The flash accelerator is implemented through a Max 10 device that runs instructions directly (execute in place) from user flash memory (UFM) or a Max 10 integrated on-chip flash. The flash memory for Max 10 operates at 120MHz, producing 32-bit data each cycle. The read access to the UFM through a normal Nios II instruction master introduces five cycles of wait state each time because the Nios II instruction master does not support burst at the UFM burst boundary. The wait states refer to the number clock cycles for the data to be available at the output of the on-chip flash.

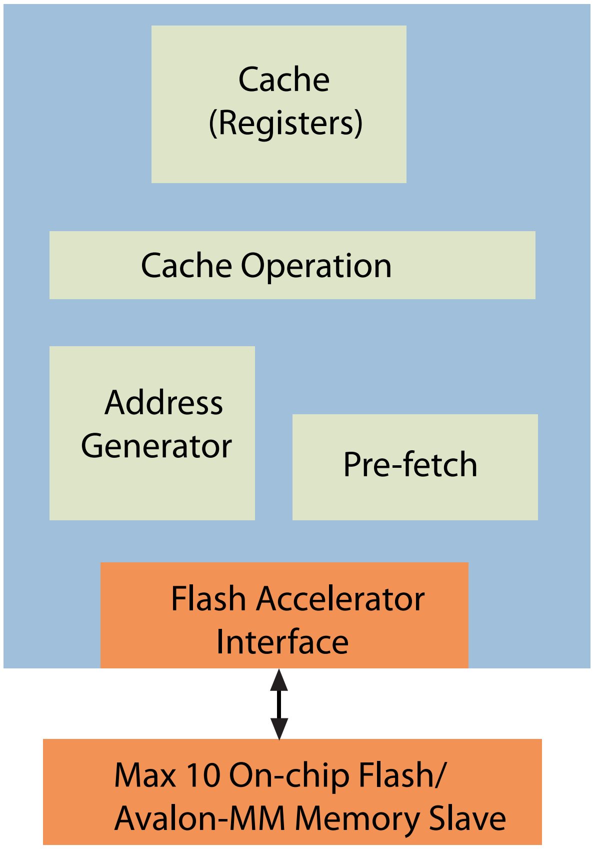
Flash accelerator takes advantage of the wait states by performing the next cache line fetch (pre-fetch) during the five cycle wait states. The on-chip flash IP latches the next read address while the read data of the previous transaction becomes available at its data registers. The valid data arriving at the flash accelerator is stored into a fully-associative cache that is implemented in registers. This speeds up instruction execution when running from high latency memory such as flash memory.

The use of flash accelerator is not limited to the Max 10 on-chip flash IP but can also be connected to other memory devices with a standard Avalon-MM master interface. It is suitable for Nios II systems that requires smaller cache and do not use any memory block.

Feature Description

The Nios II flash accelerator provides a small fully-associate cache where both its line and cache sizes are user-configurable. The cache is constructed from registers. Standard Nios II cache management instructions such as `flushi` and `initi` can be used to flush or initialize the flash accelerator cache. The flash accelerator supports sequential instruction pre-fetch to improve performance over sequential instruction execution.

Figure 1: Block Diagram of Flash Accelerator



Cache Operation Block

This block detects cache misses and triggers read requests to the Address Generator. It also handles cache fill operations. Incoming read data are filled into the cache lines using the Least Recently Used (LRU) technique to ensure that the most used cache lines remain in the cache.

It handles Nios II cache management instructions such as `flushi` and `initi`.

Pre-fetch Block

The Pre-fetch block contains circuitry that detects the next sequential cache line to be executed relative to the current executed cache line. If the next line is predicted to miss and that there is no cache operation being performed (idle mode), the Pre-fetch block can trigger a read request to the Address Generator.

Address Generator Block

This block handles read requests from the Cache Operation and Pre-fetch blocks and generates the Avalon-MM read transaction via the Flash Accelerator Interface.

Cache (Registers) Block

The two main cache components of this block, the cache tag and cache data are composed of registers. The cache tag stores part of the address while the cache data stores valid read data. The cache data size is configurable through Qsys parameters.

Interfaces

It is a 32-bit read-only Avalon-MM master interface. It has support for wrapping burst where the burst starts from the critical word first. The burst size is configurable through the **line size** parameter where the burst count size equals to Line Size divided by 32-bits data.

Note: The Max 10 on-chip flash IP "Read burst mode" parameter needs to be configured to wrapping burst.

Parameter

The table below lists the user parameters to allow the Flash Accelerator to be configured via the Nios II Qsys GUI under the "**Cache and Memory Interfaces**" tab. They are only available when the Nios core is configured to be Nios II/f.

Table 1: Flash Accelerator Qsys Parameters

Parameter	Usage	Configurable Option
Line Size	Determines the width of each cache line in bits: `Burstcount size = Line Size (bits)/32`	• None • 64 bits • 128 bits
Cache Size	Determines the number of cache lines for the cache data	2, 4

Performance

The read latency shown in the following table is estimated based on 10M08, 10M16 and 10M25 User Flash Memory devices that have five cycles random read and is compared between a normal Nios II instruction master and flash accelerator interface.

Table 2: Max 10 on-chip flash IP Read Latency

Nios II Instruction Interface	Random Access	Sequential Access
Normal	5	5
Flash Accelerator	5	1

For a normal instruction master, fetching sequential addresses from the Max 10 On-Chip Flash IP is considered a random access because it does not take advantage of the burst read feature of the Flash IP by default. Enabling the optional `burstcount` signal at the Instruction Master does not improve the sequential access because the wrap burst of 8 from the instruction Master will be translated to equivalent incremental burst of 2 or 4 at the Flash IP. The most optimum setup is to ensure that both the master and slave have the same burst count size.

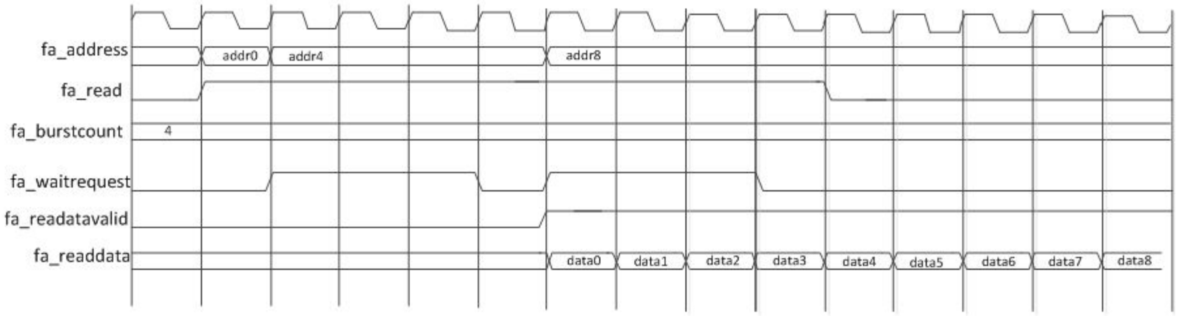
The flash accelerator has a cache line size that can be configured to match the corresponding Max 10 on-chip flash burst size. It is recommended that the flash accelerator line size is configured according to the Max 10 device family line sizes listed below for better performance.

Table 3: Cache Line Size Settings

Max 10 Device	Cache Line Size (bits)
10M08	64
10M16	128
10M25	128
10M50	128

The flash accelerator also has pre-fetch feature that allows the sequential cache line to be filled to take advantage of the wait states. The timing diagrams for reads at flash accelerator is shown in the following figure:

Figure 2: Timing diagram for Flash Accelerator for 10M16/10M25



Document Revision History

Date	Version	Changes
June 2015	2015.06.30	Initial release