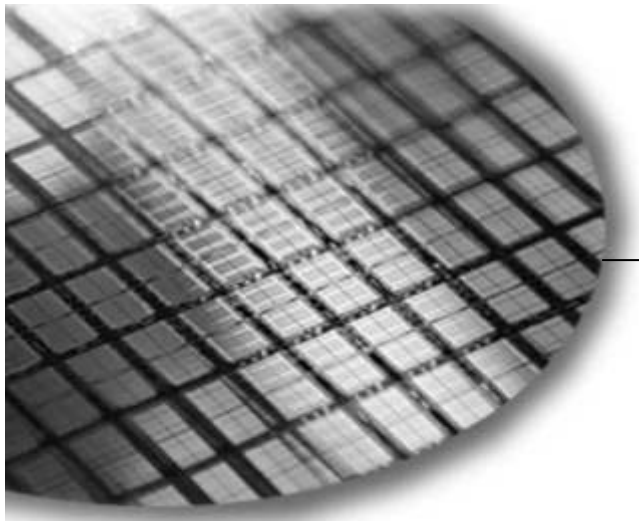




The World Leader in High Performance Signal Processing Solutions



Connecting the ADV202 and Blackfin Considerations and Pitfalls

Boston FAE Training
Steve Ranta, NW Linear FAE
March 7, 2006

ADI Confidential Information – Not for external distribution



Why do I get to present this subject?

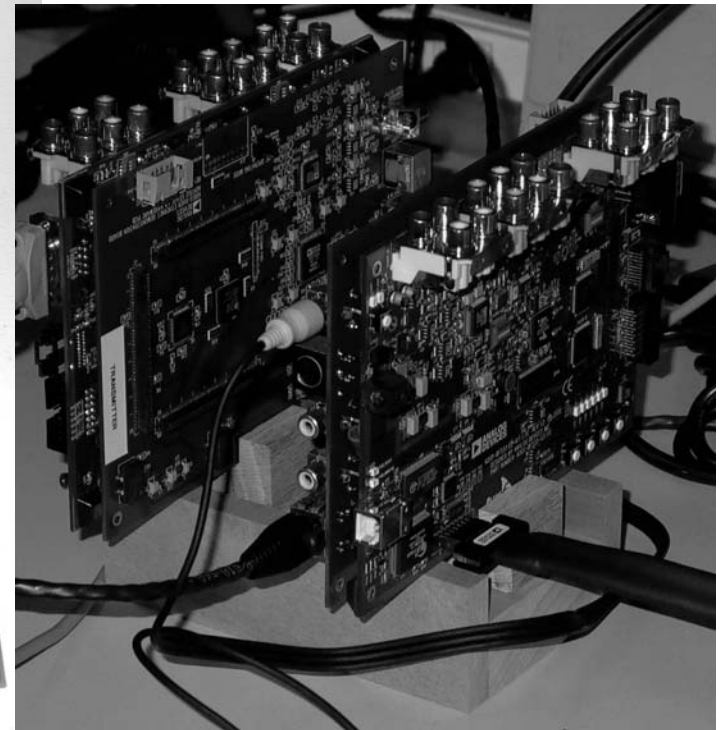
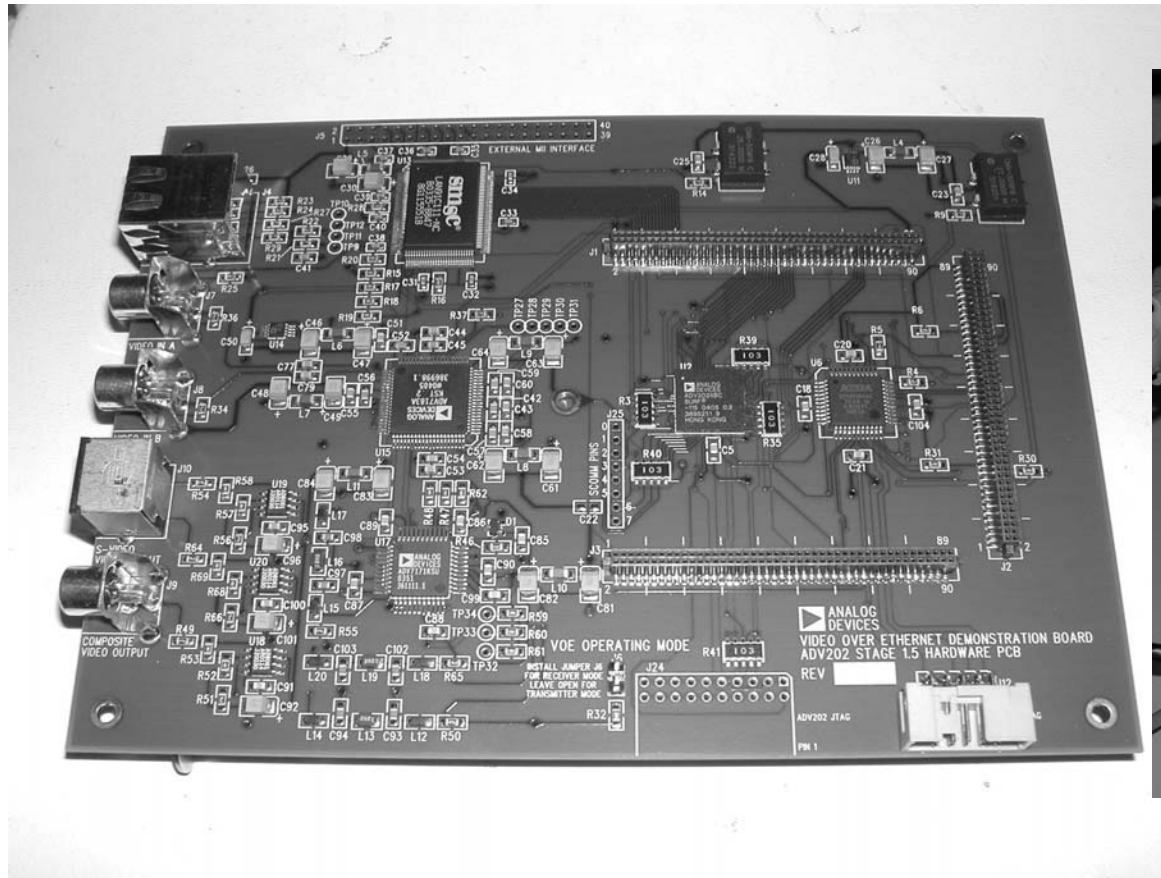
- ◆ **Four years as an Analog Devices FAE working with customers in analog and digital video**
 - Past design engineer experience working in analog video systems
- ◆ **Two years of experience in working with several high profile customers using the ADV202**
- ◆ **Recognized the advantages of using JPEG2000 compression technology in video applications**
 - Also realized how the ADV202 encapsulated the technology and made it (relatively) easily accessible to our customers
- ◆ **I developed a “real world JPEG2000 video application” demonstration using the ADV202 called “The VOE Project” (2003-2004)**
 - Customers were not familiar with JPEG2000, had bias towards MPEG but liked the low latency and error resiliency aspects of JPEG2000 and the ability to do HD encode or decode in a two chip solution
 - Customers wanted to see an example design of an embedded video over IP network platform using the ADV202 before jumping on the JPEG2000 bandwagon
 - No standalone (non-PC) reference design existed up to that time to demonstrate such a solution

VOE Project

What was it?

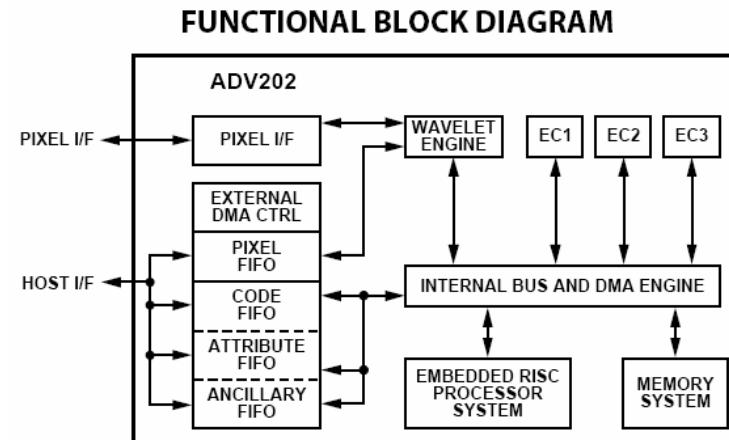
- ◆ **VOE=Video Over Ethernet**
- ◆ **A SD video over IP example design using ADI solutions**
 - The goal was to gain understanding and become intimately familiar with the ADV202
 - ADI had solutions for all major HW sockets except the memory, configurable logic, and the Ethernet MAC/PHY
- ◆ **Use available HW tools from ADI to get up to speed quickly**
 - Needed a capable processor to manage both ends of the link
 - Blackfin 533 EZKit provided most of the digital side of the system and was ready to use out of the box
 - EZKit provided an expansion interface to plug in daughter cards that allowed customized hardware to interface with the Blackfin
 - The VOE Daughter card provided analog video decoder and encoder, ADV202, and Ethernet MAC/PHY functions

VOE Project Type B EZKit Daughter Card



Architecting the VOE

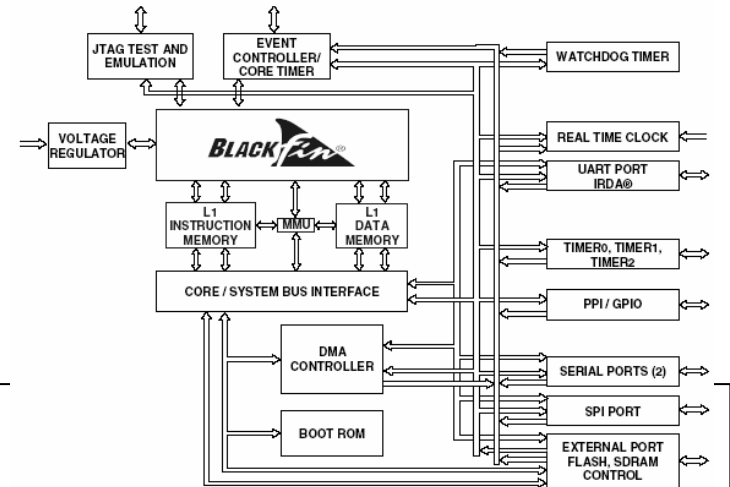
ADV202 Perspective



- ◆ **ADV202 is a “everything but the kitchen sink” device**
 - Extremely flexible in terms of how to get data into and out of the device
- ◆ **Video I/O-VDATA port (Pixel I/F in above diagram)**
 - ◆ Directly suitable and gluelessly connects to “656” style interfaces for video decoders/encoders and HDMI/DVI interface chips
- ◆ **Host Data I/O (HOST I/F in the above diagram)**
 - **HDATA interface-16 or 32 bit wide generic memory mapped interface (SRAM style)**
 - ◆ Remember that the ADV202 is a 32 bit device internally
 - **JDATA interface-8 bit wide streaming interface**
 - ◆ Uses some of the multiplexed HDATA pins for JDATA
 - ◆ Restricts HDATA access to 16 bits only (still need HDATA when using JDATA)
- ◆ **HDATA can be used to pass compressed data, but it is always used for configuration and control of ADV202**

Architecting the VOE

Blackfin 533 Perspective



◆ Blackfin has a rich set of peripherals

● Parallel Peripheral Interface (PPI)

- ◆ Directly suitable and gluelessly connects to “656” style interfaces for video decoders/encoders and HDMI/DVI interface chips
- ◆ 66 MHz maximum clock rate (cannot be used to clock in uncompressed HD video data-requires 74.25MHz interface)
- ◆ Possible interface to JDATA compressed streaming interface on the ADV202

● SPORTs

- ◆ Not really useful to the ADV202

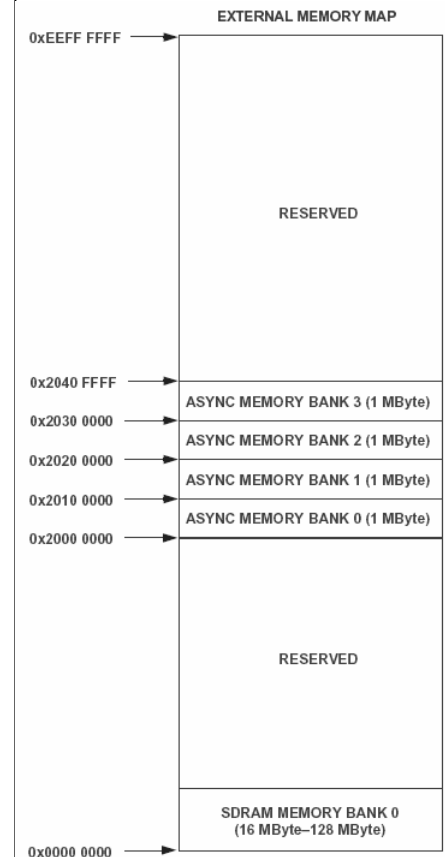
● External Bus Interface Unit (EBIU)

- ◆ On the BF533 (and other lower cost Blackfins,) the EBIU is 16 data bits wide
- ◆ On the BF561 the EBIU is 32 data bits wide
 - This BF561 was not available when the VOE was first architected

Architecting the VOE

Using the Blackfin to configure and control the ADV202

- ◆ **ADV202 requires configuration and control**
 - This must occur over the HDATA interface (16 or 32 bits wide)
 - The ADV202 is usually configured on power up
 - Control operations typically require brief reads and writes during compression or decompression
- ◆ **VOE based on the lower cost BF53x Processor**
 - Only 16 bit wide external memory interface available
 - HDATA accessed in 16 bit mode
 - ◆ Two 16 bit accesses are required to read or write registers in the ADV202
- ◆ **ADV202 HDATA interface looks like an asynchronous memory device to the Blackfin EBIU**
 - HDATA DATA[15..0] pins connected to EBIU DATA[15..0] pins
 - HDATA ADDR[3..0] pins connected to EBIU ADDR[4..1] pins
 - HDATA /RD,/WR pins connected to EBIU /ARE,/AWE pins
 - HDATA /CE should be address decoded through logic or tied to one of the /AMS[0..3] pins on the EBIU





Architecting the VOE

The compressed data path between the ADV202 and the Blackfin

- ◆ **There are two choices in how compressed data gets in and out ADV202**
 - **Through CODE FIFO register reads and writes over the HDATA interface (16 or 32 bits wide)**
 - **Through the JDATA streaming interface**
 - ◆ 8 data bits wide with one data valid and one control signal
 - ◆ Synchronous with MCLK of the ADV202
 - ◆ JDATA pins are multiplexed with the upper 16 bits of the HDATA interface so only 16 bit HDATA accesses are possible when using the JDATA interface
- ◆ **From a data traffic management perspective, JDATA would be ideal for the compressed data path between the ADV202 and Blackfin PPI**
 - **In the VOE, the EBIU is connected to SDRAM, FLASH, and Ethernet MAC/PHY, and ADV202; as such this bus is highly utilized**

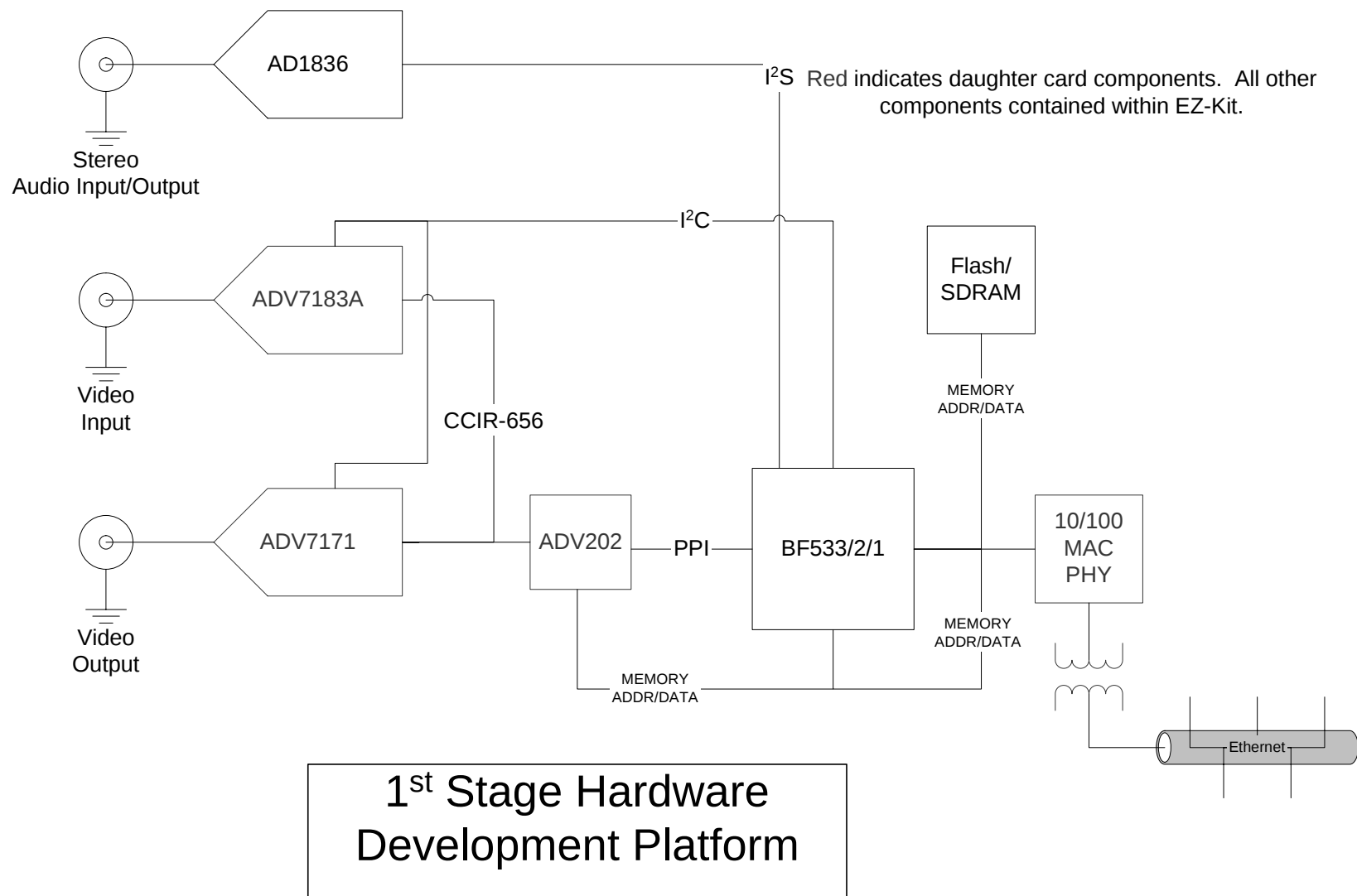
Architecting the VOE

Understanding the PPI on the Blackfin

- ◆ **8-16 bit wide parallel interface (choice of width through register configuration)**
 - Data clocked in on PPI_CLK (choice of edge through register configuration)
 - Captured data must have a DMA set up to direct data to a storage location
- ◆ **PPI primarily designed for uncompressed video data transfer**
 - **PPI 2D DMAs are designed to transfer fixed-size defined blocks of data**
 - ◆ Uncompressed video has defined characteristics (horizontal pixels per line, vertical lines per frame) that are ideal for 2D DMA transfers
 - **1D DMAs are used in GP PPI Modes**
- ◆ **PPI data transfer must be initiated by a “trigger”**
 - Configurable through PPI registers
 - Can be software initiated
 - Frame sync pins can trigger capture or output

VOE Project-The Original Architecture

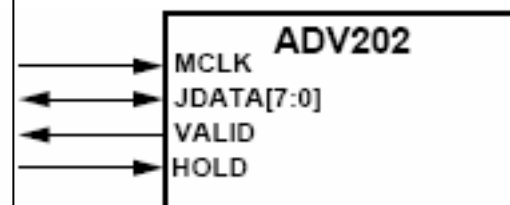
No audio functionality has been implemented in the
VOE software at this time



Architecting the VOE

Using the JDATA on the ADV202 to interface with the PPI on the Blackfin

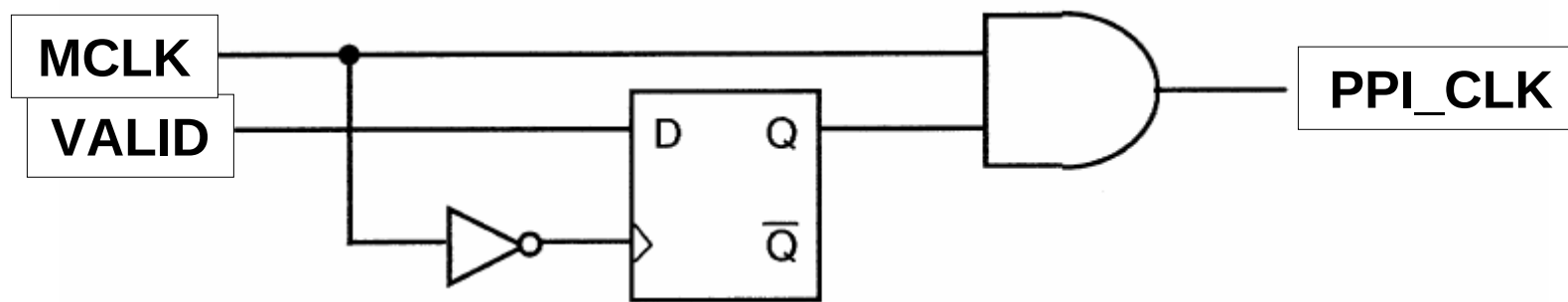
- ◆ **On the surface, the JDATA-PPI interface looks ideal**
 - JDATA port is updated synchronously on ADV202 MCLK
 - PPI interface is updated synchronously on PPI_CLK
 - Why not connect the two together?
- ◆ **Unfortunately, JDATA data isn't always valid on every MCLK when compressing video**
 - Due to how the ADV202 works on the video data, the compressed data is not produced at a constant rate from moment to moment-therefore it is "bursty" in nature
 - VALID pin on JDATA frames when data is "good"
 - ◆ the status of this pin is not deterministic on a byte to byte basis-must use to frame good data
- ◆ **In order to clock good (valid) data into PPI interface, a method to stall the PPI was envisioned to clock in only good data**
 - What about gating the PPI_CLK with VALID?



Architecting the VOE

Gating the PPI_CLK to stall the PPI on the Blackfin

◆ Used the following circuit to gate the PPI_CLK



- ◆ JDATA is updated on MCLK but good data is framed by VALID
 - Utilizing this clock gating circuit allowed the VALID bit to gate PPI_CLK
 - The idea is that the PPI could now only capture good (valid) JDATA
- ◆ Unknown to most people at the time of the VOE (2003), the PPI does not work correctly with a gated PPI_CLK
 - Internal clock domain synchronizer needs uninterrupted clock
 - ◆ PPI data must be resynchronized from the PPI_CLK domain to the SCLK domain in the Blackfin
 - Data will get corrupted for the first 4 or 5 PPI_CLKs when clock is applied
 - Gating the PPI_CLK with VALID won't work for JDATA-PPI interface

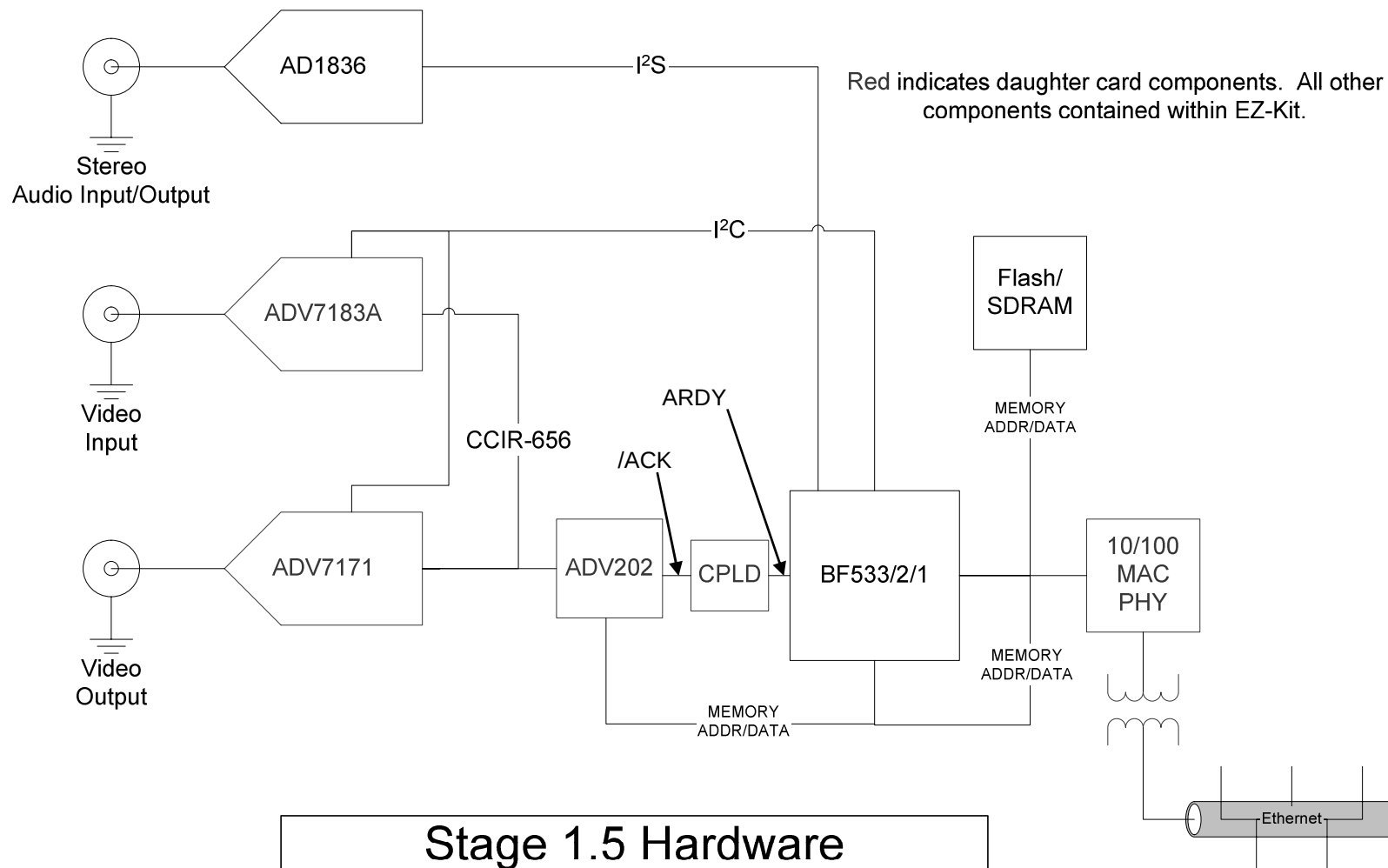
Architecting the VOE

Going to Plan B

- ◆ **It took a fair amount of time to understand and isolate that stalling the PPI_CLK caused the corruption of the PPI data transfer**
 - **Due to time deadlines, it was decided to implement Plan B**
 - **Plan B was to transfer compressed data over the existing HDATA interface on the external memory bus**
- ◆ **Using HDATA for compressed data meant putting more data on an already congested external memory bus**
 - **SDRAM**
 - ◆ Although instruction caching was used, program and data were stored in SDRAM, utilizing a fair share of the external bus bandwidth
 - **Ethernet MAC/PHY**
 - ◆ Maintaining Ethernet packets used a fair chunk of the external bus bandwidth
 - **FLASH**
 - ◆ Intermittently accessed on power up and when certain data needed to be stored in a non-volatile manner, uses very little bus bandwidth
- ◆ **Throughput of the system is now determined by the external bus bandwidth, not processor MIPS**

VOE Project-The Current Architecture

No audio functionality has been implemented in the VOE software at this time

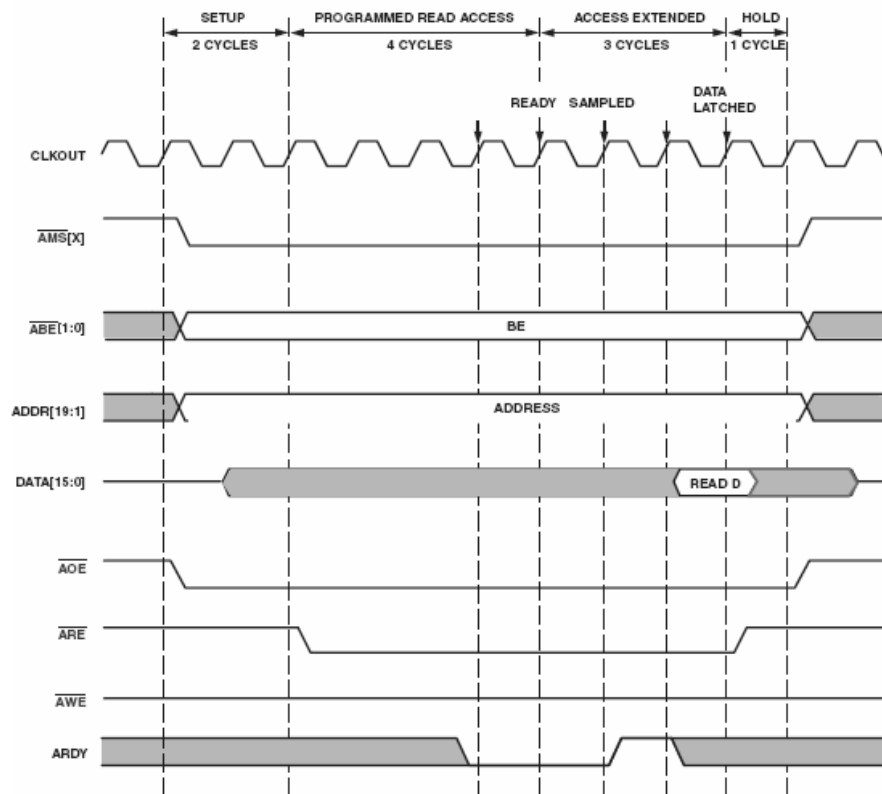


Stage 1.5 Hardware Development Platform

Architecting the VOE

Understanding the Blackfin EBIU transactions

- ◆ **Blackfin EBIU reads and writes are divided up into four phases: setup, data access, data hold, and bus transition time**
 - Each phase duration is configured as some number of SCLKs
 - **ARDY pin can be used to extend the data access phase indefinitely**
 - ◆ This is useful when a memory mapped device needs to insert extra wait states to put valid data on the bus
 - **ADV202 HDATA interface has dramatically different data access times depending on whether the host processor requests direct or indirect memory access within the ADV202**
 - ◆ A method to utilize ARDY is required to keep the HDATA data access time to a minimum, keeping bus efficiency maximized



Architecting the VOE

Understanding the ADV202 HDATA transactions

- ◆ **ADV202 asserts /ACK when HDATA data is valid on access**
 - **An inverted version of /ACK (gated with /CS) looks like it could be used to drive ARDY directly**

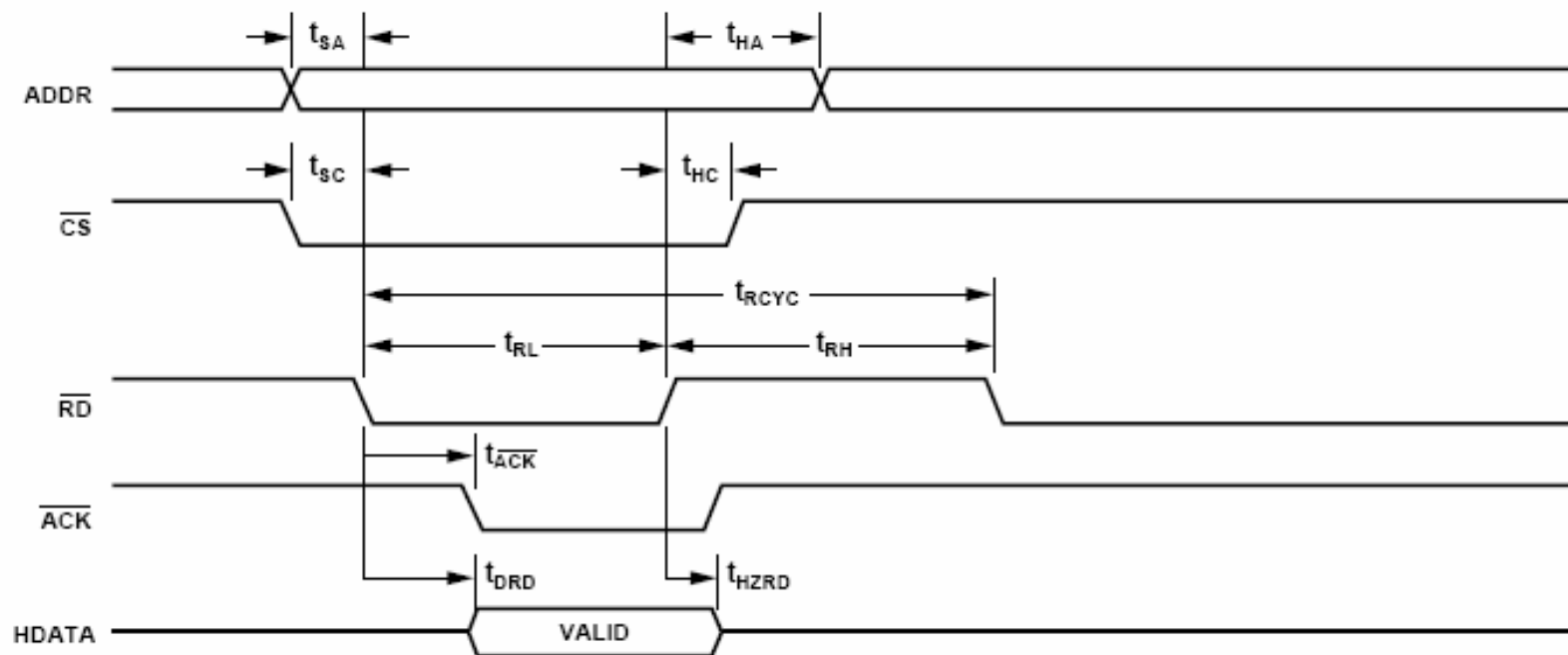


Figure 3. Normal Host Mode—Read Operation

Architecting the VOE

Using the ADV202 /ACK as the Blackfin EBIU ARDY (Note: Not tested!)

- ◆ **ADV202 must configure /ACK as open drain**
 - This is not the default state of /ACK
 - Must pull up /ACK-ARDY signal with an external resistor
- ◆ **Blackfin ARDY enable and active polarity is configurable by bank**
 - Enable ARDY for the bank that is memory mapped to the ADV202
 - ARDY assertion polarity defaults to active high, need to set for active low
- ◆ **EBIU read and write access time is configurable by bank**
 - Set R/W access time to the minimum value to force the extension of the EBIU access phase to terminate on the /ACK signal
 - ◆ This allows the fastest reads and writes to occur on the ADV202 whether or not the access is to direct or indirect registers
 - ◆ Enable watchdog timer to force a Blackfin hardware reset in case the ADV202 does not /ACK due to hardware error
 - This will prevent an indefinite bus hold and allow the system to recover from an ADV202 lock up

Final VOE Project Observations

Maximizing throughput on the external memory interface

- ◆ **VOE utilized off the shelf RTOS to manage TCP/IP and schedule program tasks**
 - RTOS we utilized had an inefficient RTP stack
 - RTP stack forced copying of packet payload data, stealing precious bandwidth from the bus to perform these tasks
 - ◆ Look for stacks that DMA payload data to memory and then give a pointer to that data
- ◆ **Revisit using the PPI for consuming compressed JPEG2000 data (not tested)**
 - Relieves traffic congestion on the external memory bus
 - During compression, clock JDATA data bits plus the VALID bit as 9 bit data
 - Clock PPI_CLK with ADV202 MCLK
 - After DMA into internal memory, parse data for VALID bit and cast off invalid data
 - ◆ May need to keep track of number of invalid data samples for decompression purposes
- ◆ **Using PPI to output data to JDATA during decompression may be a bit trickier as ADV202 can try to “hold off” incoming data (not tested)**
 - PPI output 8 bit compressed data plus VALID bit
 - JDATA has a flow control mechanism (HOLD) to try to regulate the flow of the data source (the PPI) if the CODE FIFO becomes full
 - PPI has no ability to hold off data so VALID data must be interleaved with an appropriate amount of invalid data as a crude form of flow control so that CODE FIFO doesn't overflow (and HOLD never gets asserted)
 - ◆ Knowing number of invalid samples counted on the transmission side can help implement the decompression flow control mechanism on the Blackfin



Presented By:
Steve Ranta
NW Linear FAE



Analog Devices, Inc.
11225 SE 6th Street
Bellevue, WA 98004
PHONE 425-698-4919
steve.ranta@analog.com