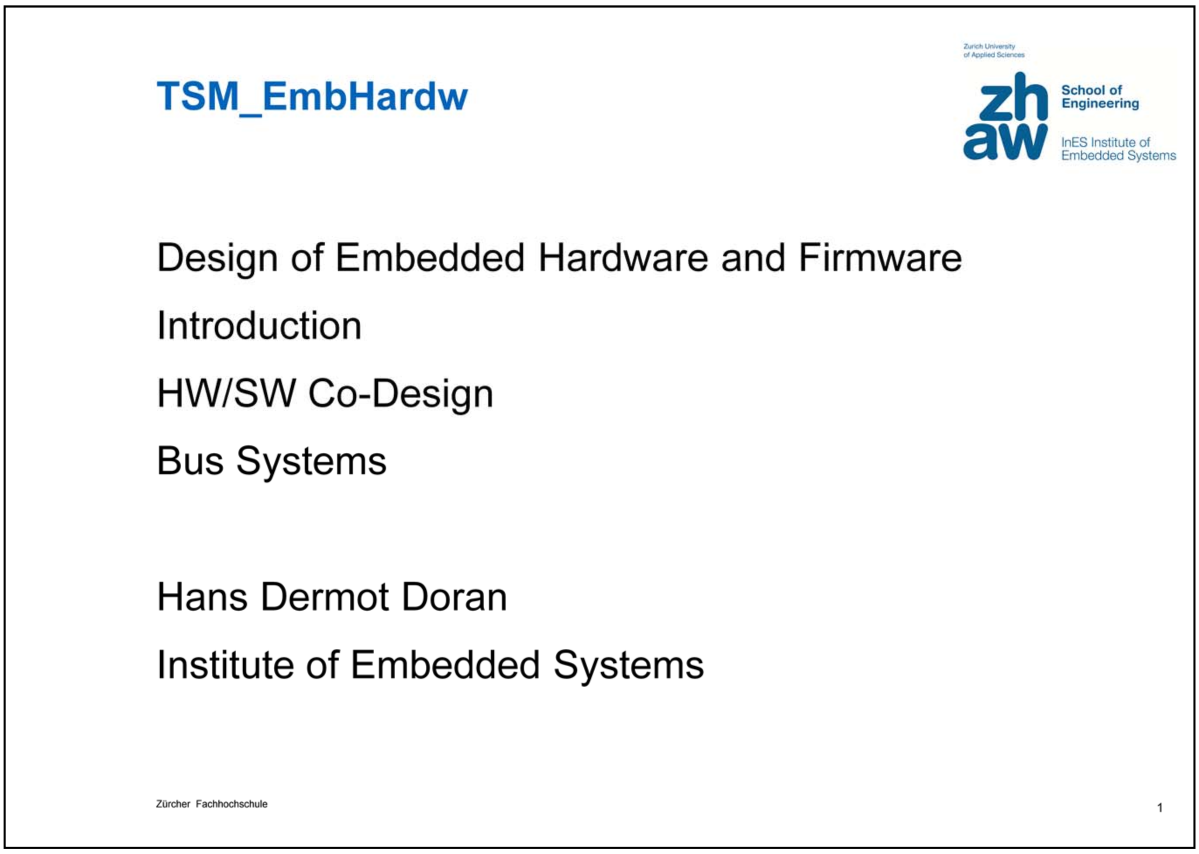


TSM_EmbHardw

Zurich University of Applied Sciences

zh aw School of Engineering

InES Institute of Embedded Systems

Design of Embedded Hardware and Firmware

Introduction

HW/SW Co-Design

Bus Systems

Hans Dermot Doran

Institute of Embedded Systems

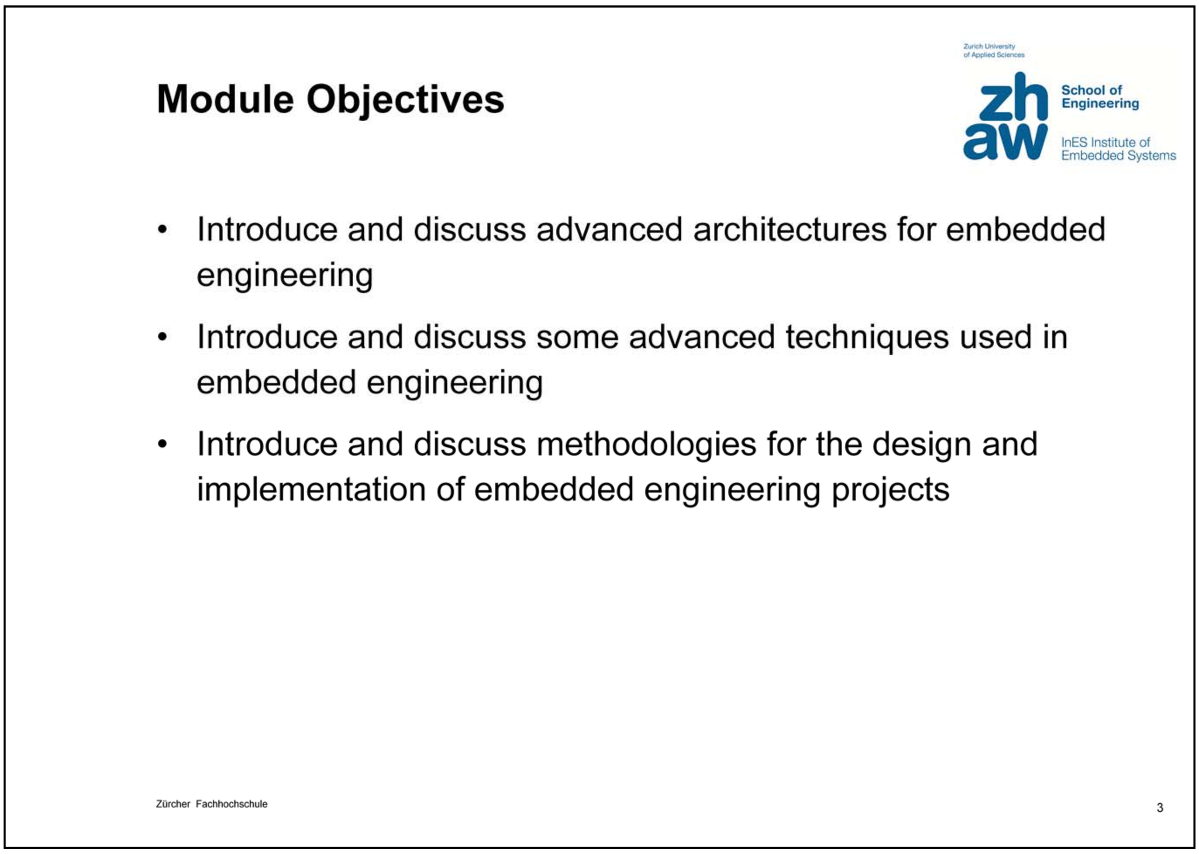
Zürcher Fachhochschule

1

V1.0	07.11.11	Initial version
V1.01	14.11.11	Corrected front page
V1.02	01.11.12	Minor overhaul – added agenda slide
		Re-wrote slides 3-5
V1.03	06.04.13	Minor edits
V1.04	20.03.15	Major edits
V1.05	18.04.16	Integrated bus systems

Agenda Module

- Module Objectives
- Second Part Objectives and Organisation
- Lecture Objectives



Module Objectives

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems

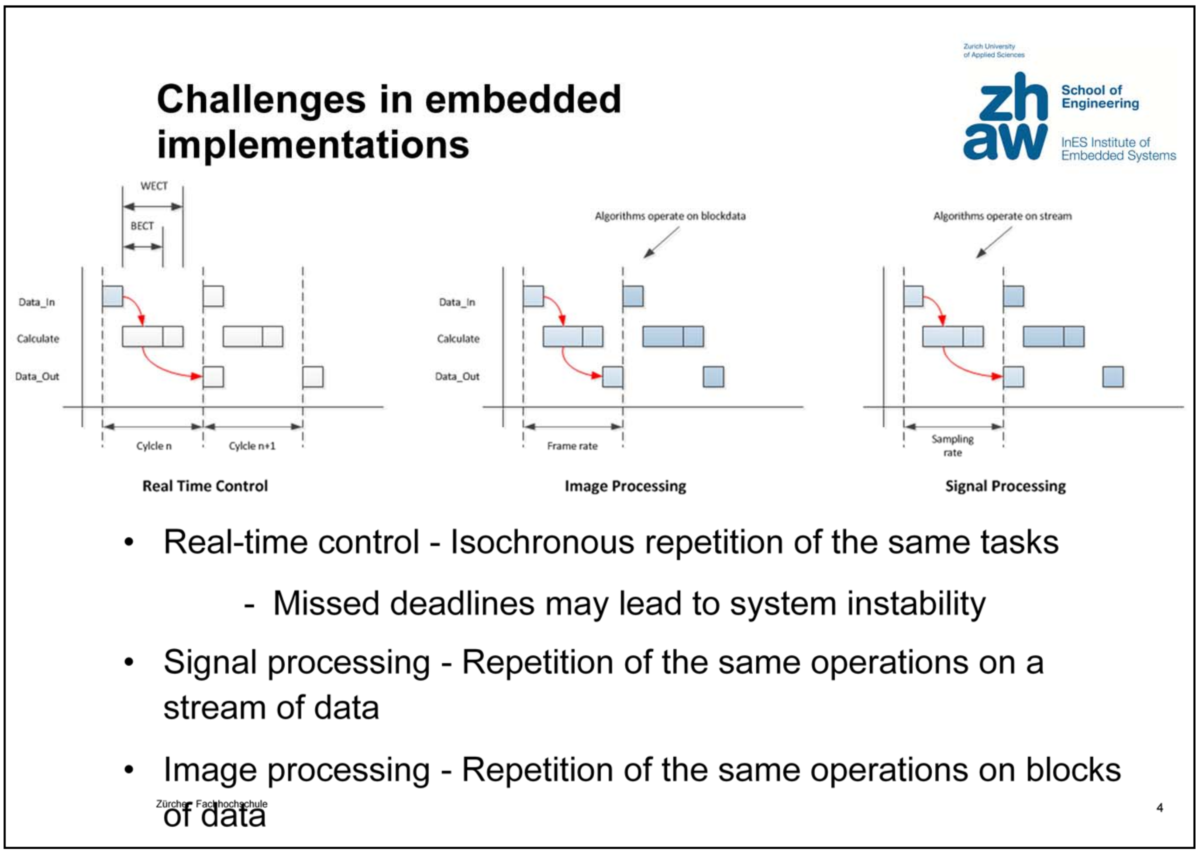
- Introduce and discuss advanced architectures for embedded engineering
- Introduce and discuss some advanced techniques used in embedded engineering
- Introduce and discuss methodologies for the design and implementation of embedded engineering projects

Zürcher Fachhochschule 3

The primary objectives of the module are to introduce the students to some techniques, architectures and methodologies for new-generation embedded engineering implementations

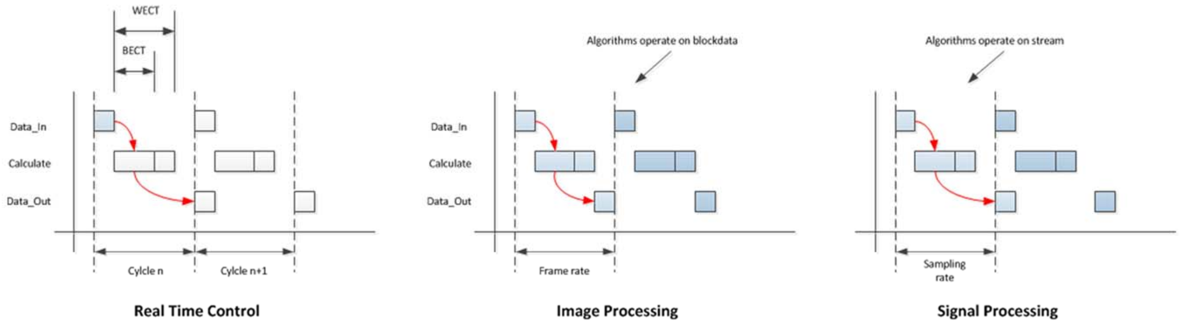
To achieve this the know-how in the class must be homogenised. Rather than doing this in class it is expected that the students prepare themselves adequately.

Of necessity some of the material handled has already been covered at bachelor level but it is expected that this will be handled in more detail here.



Challenges in embedded implementations

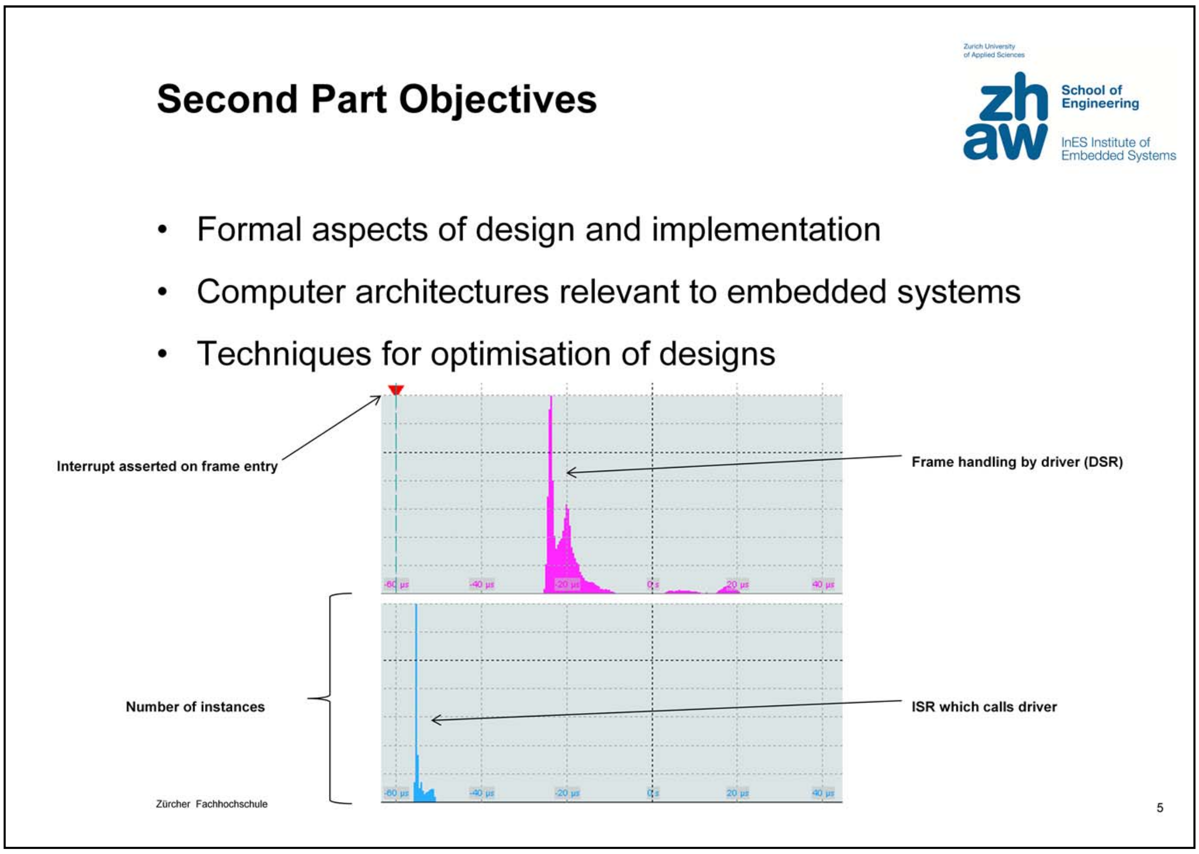
Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems



- Real-time control - Isochronous repetition of the same tasks
  - Missed deadlines may lead to system instability
- Signal processing - Repetition of the same operations on a stream of data
- Image processing - Repetition of the same operations on blocks of data

Zürcher Fachhochschule 4

Embedded Engineering is not theoretically deep – it consists largely of the use of cost optimised and purpose built computer systems. The slide shows three applications where the performance must typically be optimised. In real time control the control loop defines the cycle time and the control loop software runtime, including error handling, must be completed within the cycle time. In image processing bulk data must be fetched and processing completed before the next frame is fetched. Similarly signal processing acts on a stream. Worst Case Execution Time and Best Case Execution Time must be known for all applications.



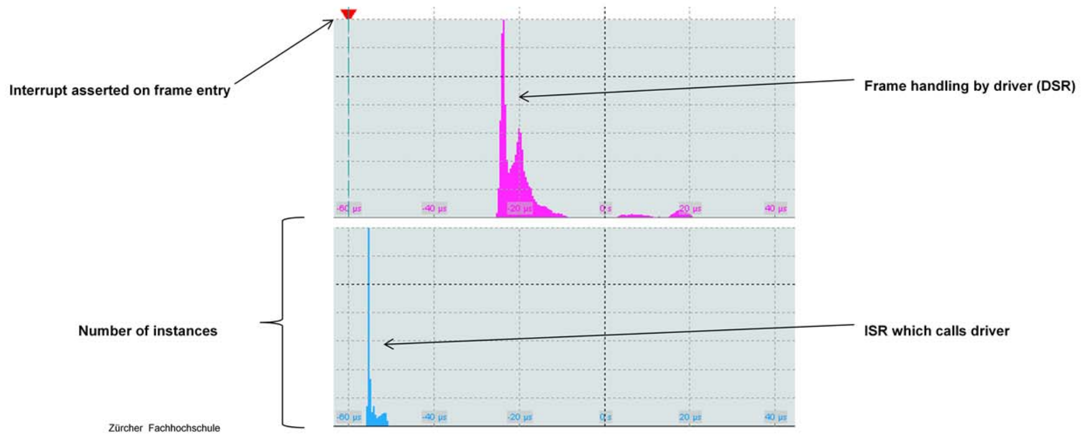
Second Part Objectives

Zurich University of Applied Sciences

zh aw School of Engineering

InES Institute of Embedded Systems

- Formal aspects of design and implementation
- Computer architectures relevant to embedded systems
- Techniques for optimisation of designs

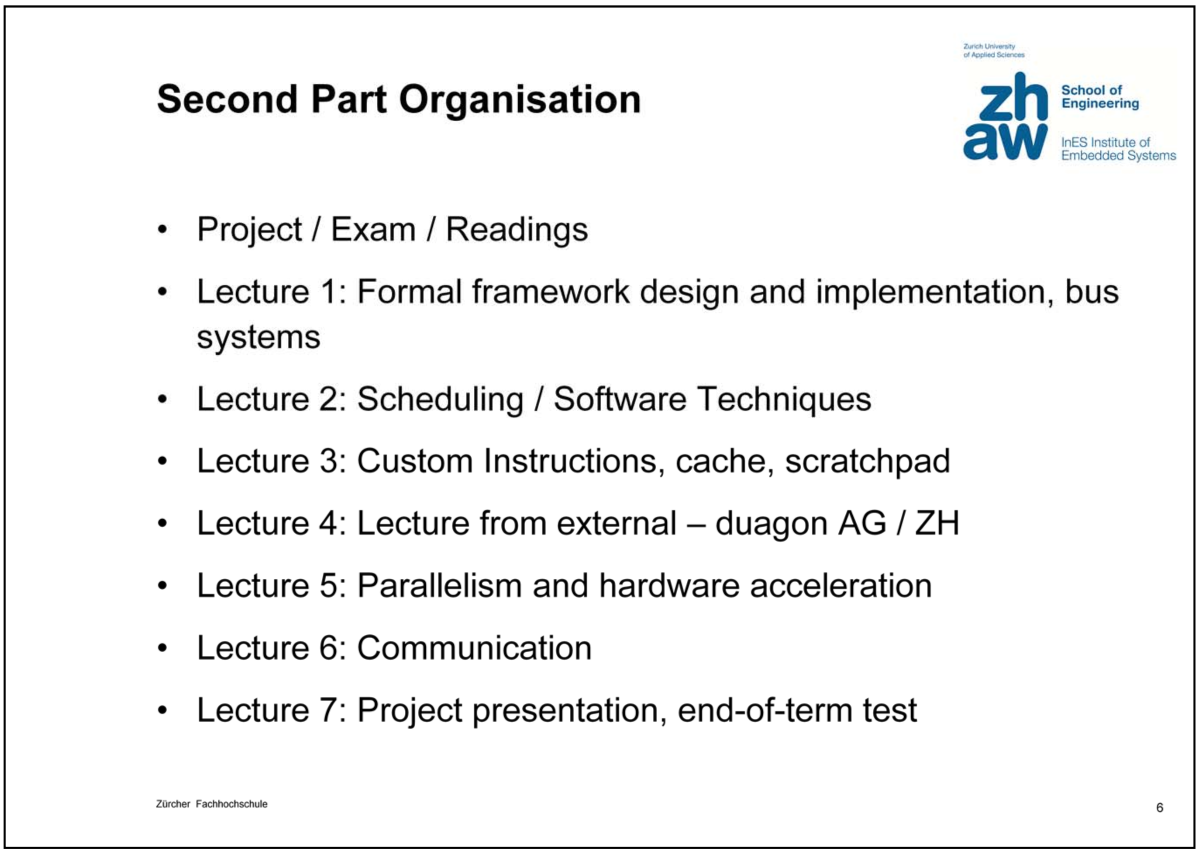


Zürcher Fachhochschule

5

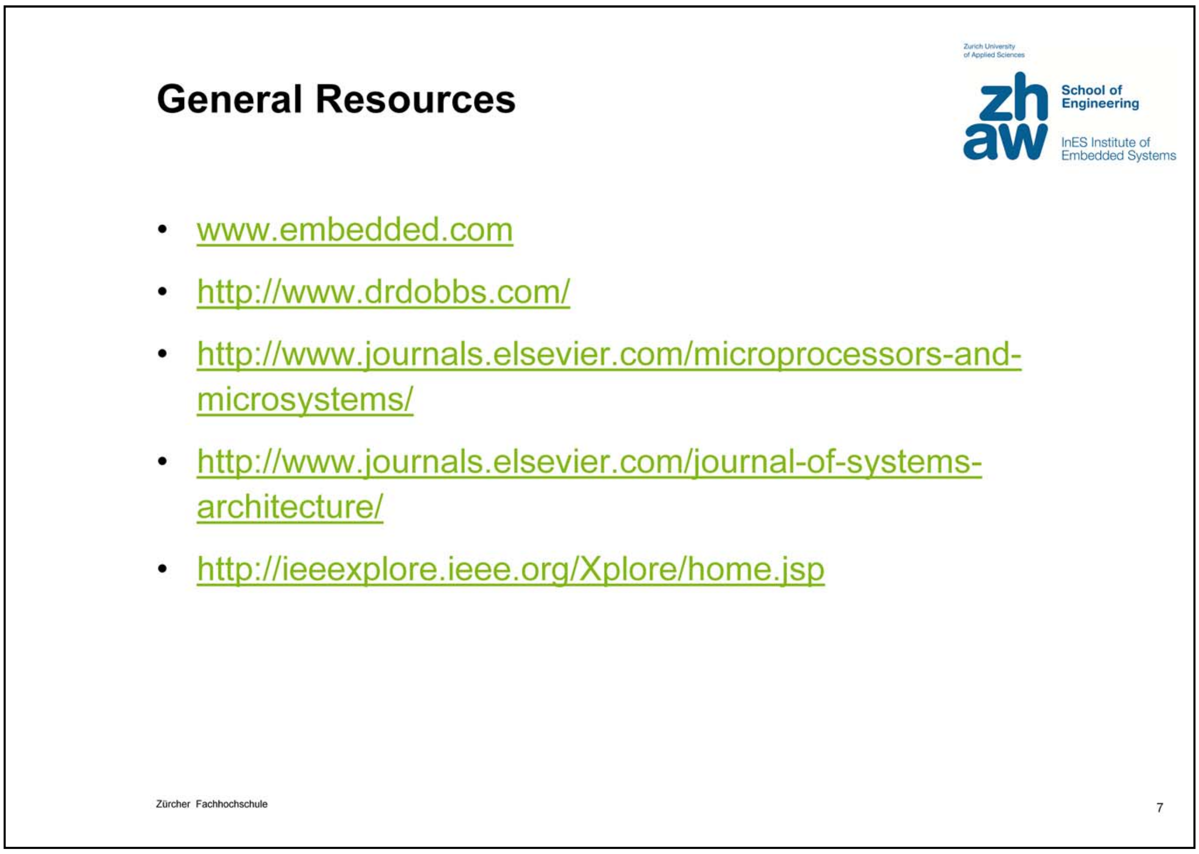
Limits of the module are described by this oscilloscope trace – The trigger is a signal that generated an interrupt, the lower trace shows the processor latency calling a single ISR which then calls an OS task, seen in the top trace. Anything “left of -55us” is out of the domain of this module.

(100MHz NIOS II running eCOS)



Second Part Organisation

- Project / Exam / Readings
- Lecture 1: Formal framework design and implementation, bus systems
- Lecture 2: Scheduling / Software Techniques
- Lecture 3: Custom Instructions, cache, scratchpad
- Lecture 4: Lecture from external – duagon AG / ZH
- Lecture 5: Parallelism and hardware acceleration
- Lecture 6: Communication
- Lecture 7: Project presentation, end-of-term test



General Resources

- www.embedded.com
- http://www.drdobbs.com/
- http://www.journals.elsevier.com/microprocessors-and-microsystems/
- http://www.journals.elsevier.com/journal-of-systems-architecture/
- http://ieeexplore.ieee.org/Xplore/home.jsp

Zürcher Fachhochschule 7

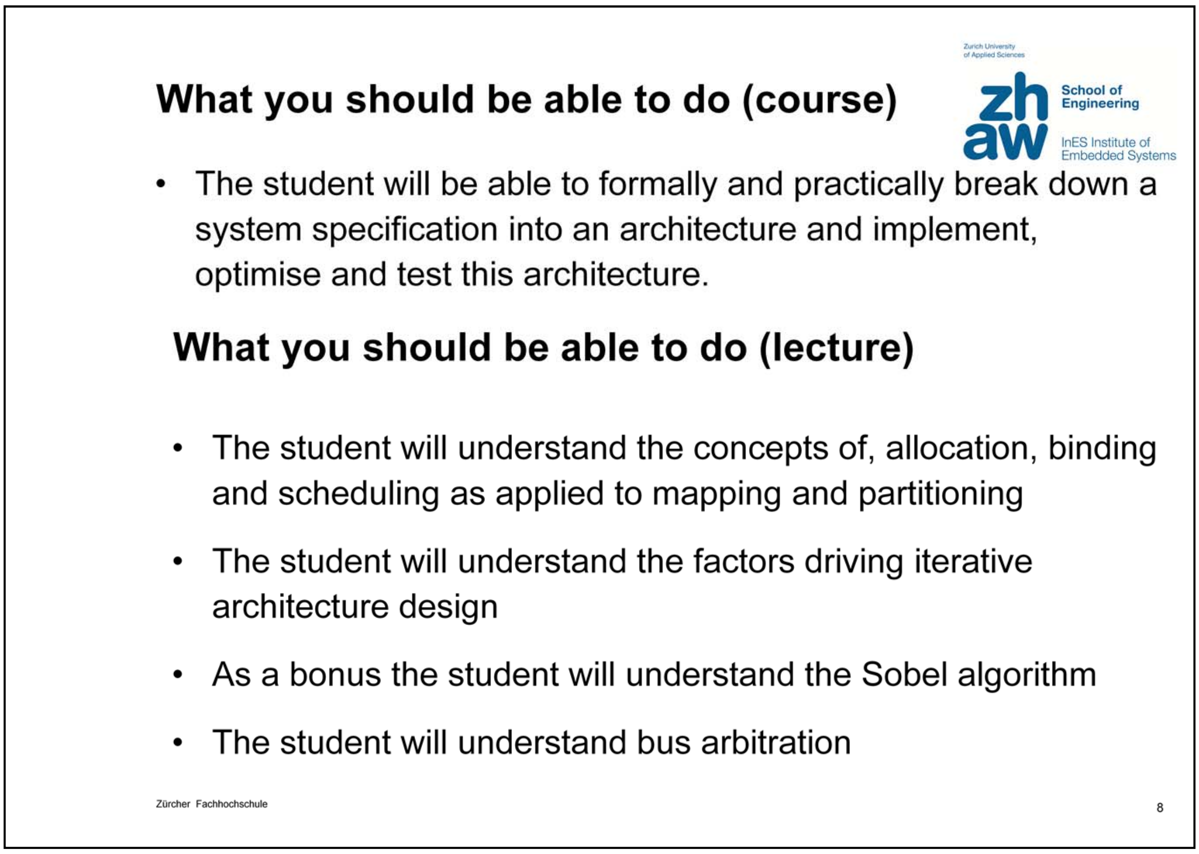
embedded.com is a useful, if not especially deep, resource for the embedded engineer. Its good for little tips on how to do things.

Dr. Dobbs is a good SW resource enclosing embedded software to PC software but rarely serious server stuff (like enterprise servers). Much of the PC stuff can also be applied to embedded systems.

The Euromicro organisation organises a couple of European conferences in the embedded/real time arena every year. It tends to be more practically applicable, i.e. a few more contributions from practitioners then the contributions published by the IEEE which tends towards the academic. They support two journals, “Microprocessors and Microsystems” and “Embedded Software Design.”

IEEE Explore hosts most of the contributions mad to their conferences and journals. Its an invaluable resource for those completing their Masters thesis.

http://www.embedded.com/design/programming-languages-and-tools/4437925/Dealing-with-memory-access-ordering-in-complex-embedded-designs-



What you should be able to do (course)

- The student will be able to formally and practically break down a system specification into an architecture and implement, optimise and test this architecture.

What you should be able to do (lecture)

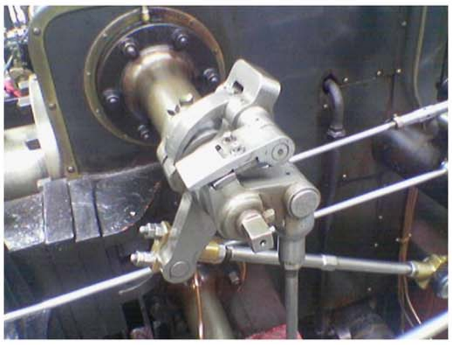
- The student will understand the concepts of, allocation, binding and scheduling as applied to mapping and partitioning
- The student will understand the factors driving iterative architecture design
- As a bonus the student will understand the Sobel algorithm
- The student will understand bus arbitration

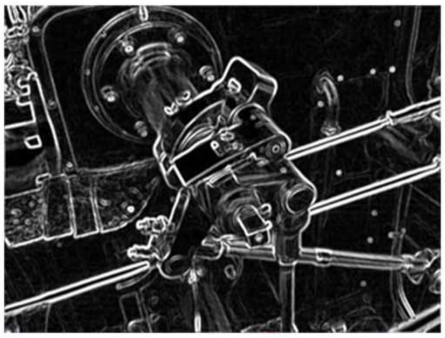
Lecture 1.1

- Working Model -> Sobel Algorithm
  - Derive the concept of mapping and partitioning
  - Derive the iterative approach of HW-SW Co-design

Practical Pragmatic Example

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems





Zürcher Fachhochschule

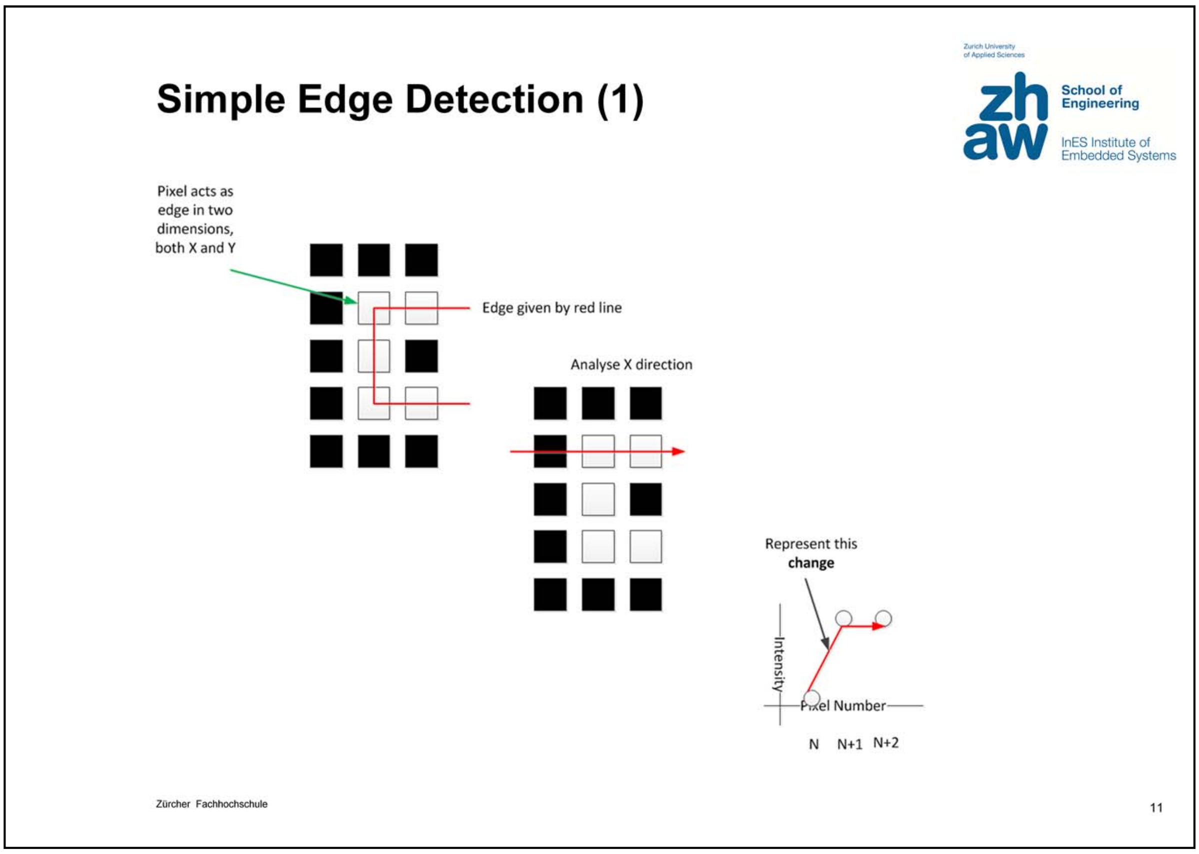
We are going to look at the challenges facing an Embedded R&D systems-engineer from a pragmatic viewpoint. What we are looking for as a demonstration model is something of sufficient computational expense to warrant analysis and optimisation. A suitable model is a common optical processing step namely edge detection. It fits well into any particular scenario for several reasons:

- It is a typical pre-processing step before more complex algorithms such as feature detection are used.
- It is computationally simple, in essence an integer vector calculation but of sufficiently computation cost to make a point.

In other words it represents an ideal candidate for optimisation.

The basic principles behind edge detection are quite simple. In our case we shall proceed to pragmatically derive the Sobel algorithm, a simple but useful algorithm.

Edge recognition attempts to identify and quantises differences in intensity in an image. Therefore it must examine the a set of pixels in order to determine whether there is any difference. Given that non-trivial images consists of multiple intensity variations some kind of **thresholding** must be applied in order to determine when an intensity variation is of interest. Equally to avoid over-computation minor intensity variations may (must) be classified as noise and eliminated from the image.

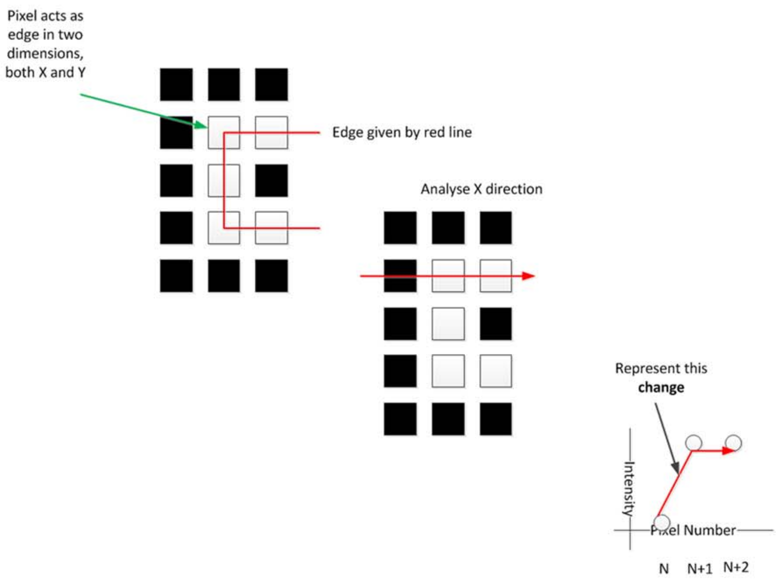


Simple Edge Detection (1)

Zurich University of Applied Sciences

zh aw School of Engineering

InES Institute of Embedded Systems



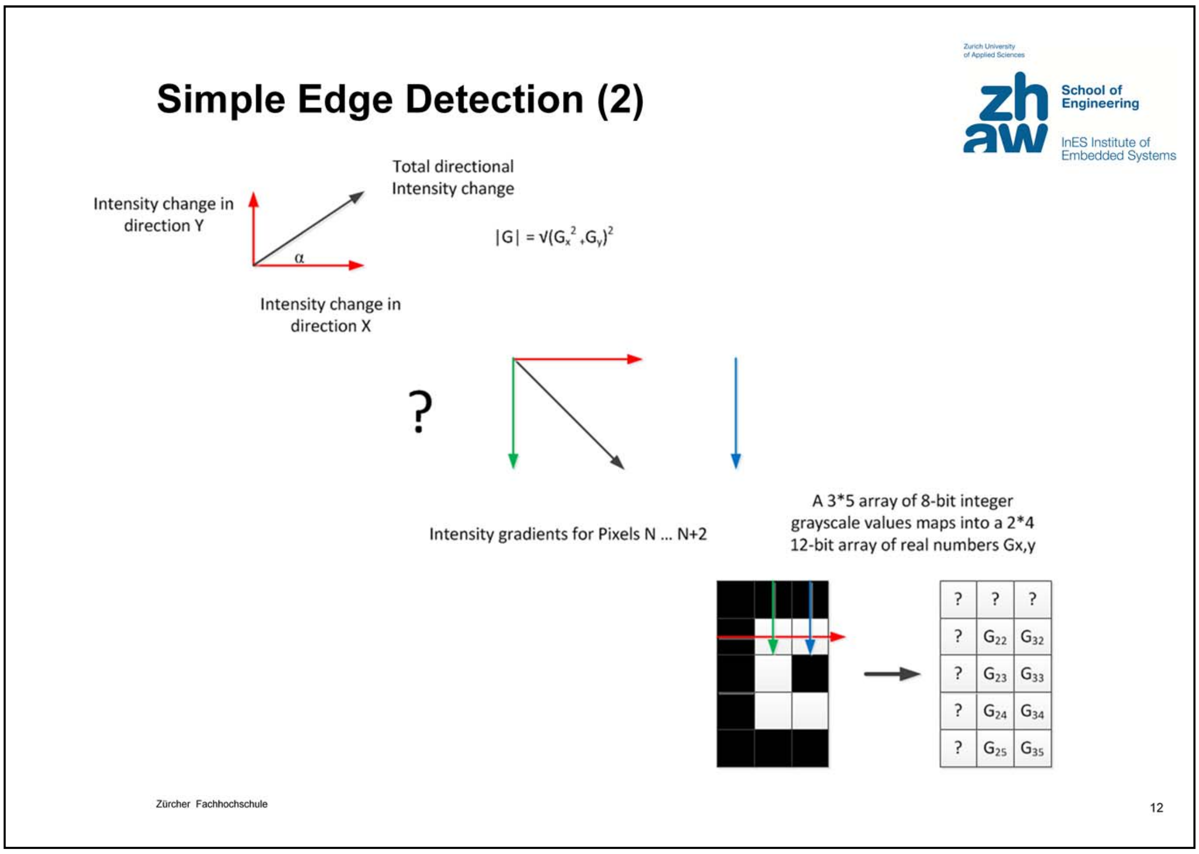
Zürcher Fachhochschule

11

Let us attempt to derive a simple edge-detection algorithm

Consider the set of pixels above – featuring part of a black and white image. The edge is in this case quite clear and denoted by the red line in the picture upper left corner. Also obvious is the concept of an edge pixel, denoted by the green arrow. This particular pixel is an edge pixel in both the X and Y directions whereas the pixel immediately to the right is an edge pixel only in the Y direction. A simple approach would be to analyse the picture twice, once for edge pixels in the X direction and once for the Y direction.

If we look at the intensity variation along the red line in the middle picture we see there is a sudden jump from the first pixel to second and no difference from that to the next. In other words we may use the rate of change as a suitable parameter to inform us that the pixel may be an edge pixel.



Simple Edge Detection (2)

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems
Intensity change in direction Y
Total directional Intensity change
α
Intensity change in direction X
?
Intensity gradients for Pixels N … N+2
A 3*5 array of 8-bit integer grayscale values maps into a 2*4 12-bit array of real numbers Gx,y

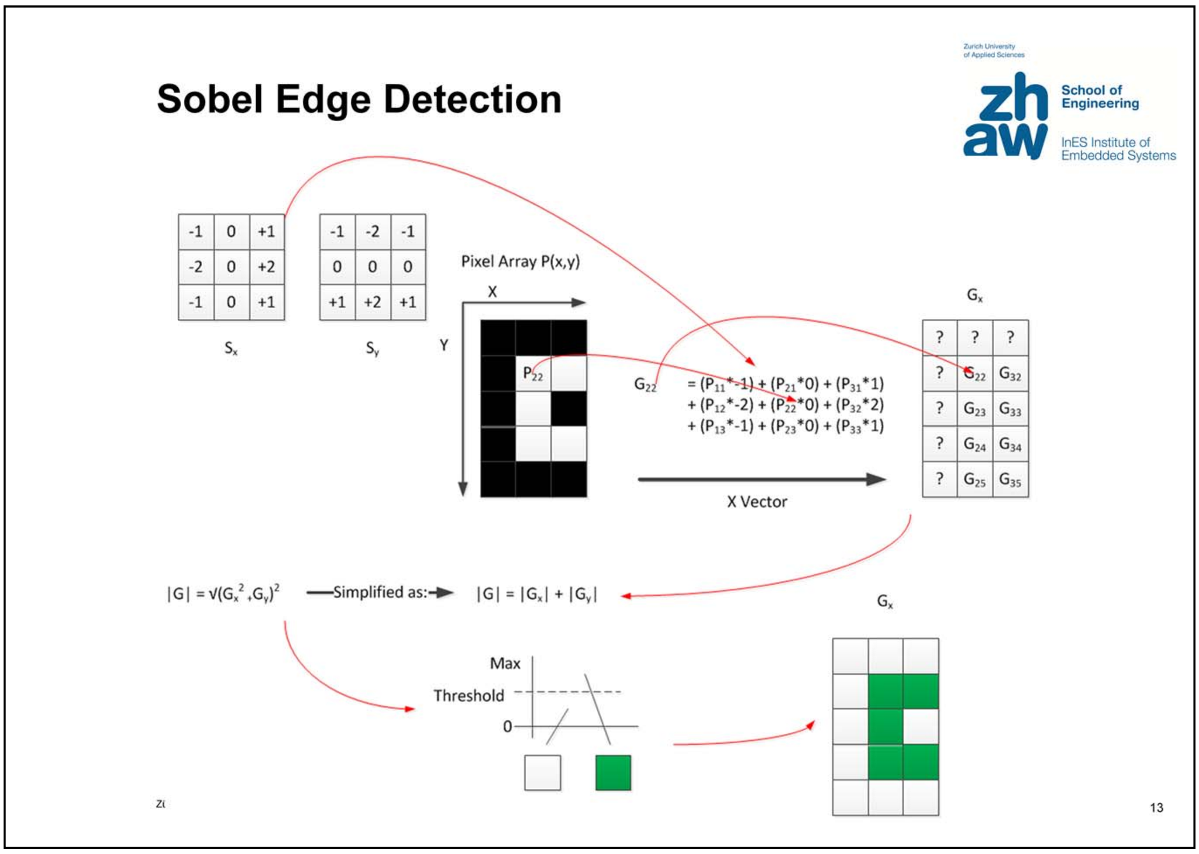
$|G| = \sqrt{}(G_x{}^2 + G_y)^2$

?	?	?
?	G_{22}	G_{32}
?	G_{23}	G_{33}
?	G_{24}	G_{34}
?	G_{25}	G_{35}

Given that we use the magnitude of the rate of change and given that we are searching along the X and Y axis for edge pixels we know we will end up with a vector, whose magnitude and direction can be calculated using Pythagoras.

A special case are the boundaries of the image – we determine the gradient of one pixel with respect to the one before it. Given that the pixels on the top and left borders of the image do not have a predecessor these cannot be defined, in the spirit of an edge detection, as edge pixels. This is represented by the question mark in the middle drawing. The vector beside it represents the change in intensity along both the X-axis and the Y axis whilst the third vector represents that along the Y axis of the third pixel – as denoted by the respective colours, red, green and blue.

One therefore ends up with – in the case of a 128 pixel by 101 line camera – a gradient array of 127 pixels by 100 lines. It is at this point also worth noting that this is not necessarily cheap on computing resources. Out of an 8 bit grayscale picture we have made a 32-bit floating value through the use of a multiplication (promotion to 16-bit) and a square root (32-bit floating point).



Sobel Edge Detection

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems
-1 0 +1
-2 0 +2
-1 0 +1
S_x
-1 -2 -1
0 0 0
+1 +2 +1
S_y
Pixel Array P(x,y)
X
Y
P_{22}
G_{22}
$= (P_{11}*-1) + (P_{21}*0) + (P_{31}*1) + (P_{12}*-2) + (P_{22}*0) + (P_{32}*2) + (P_{13}*-1) + (P_{23}*0) + (P_{33}*1)$
X Vector
G_x
? ? ?
? G_{22} G_{32}
? G_{23} G_{33}
? G_{24} G_{34}
? G_{25} G_{35}
$|G| = \sqrt{(G_x^2 + G_y)^2}$
Simplified as:
$|G| = |G_x| + |G_y|$
Max
Threshold
0
G_x
13

A more sophisticated algorithm which is simple yet produces useful results for some applications is the Sobel algorithm. In this algorithm we also distinguish between X and Y axis but we also include surrounding pixels in the analysis.

By amplifying possible intensity differences along a particular axis and also factoring in neighbouring pixels along a particular axis something approaching the probability of a pixel being an edge pixel can be estimated. Sobel does this with a so called convolution kernel with which he traverses each line of the image array, with the exceptions noted below.

The algorithm approximates $|G| = \sqrt{(G_x^2 {}_+ G_y)^2}$ with $|G| = |G_x| + |G_y|$ which serves to keep the calculations out of the floating-point processor. After determining the probabilities a decision has to be made what defines an edge and this is done with simple threshholding and from this the edges can be drawn.

Given that the gradient is calculated for pixels with 8 neighbouring pixels the gradient array of 128*101 camera reduces to an array of 126 * 99. However due to the multiplication and multiple additions there is a numerical upscaling from 8-bit to 10 bit, when programmed in software therefore to 16 bit, whole numbers.

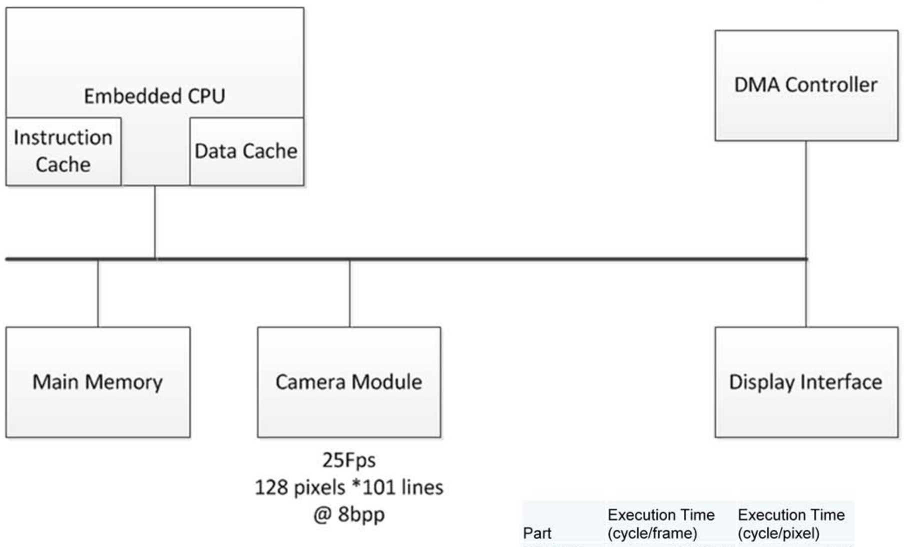
The Sobel algorithm is perfect for our analysis in that it is offers sufficient computational load and memory resources whilst appearing to be simple enough to implement in hardware.

Implementing Sobel - SW

Zurich University of Applied Sciences

zh aw School of Engineering

InES Institute of Embedded Systems



Part	Execution Time (cycle/frame)	Execution Time (cycle/pixel)
Sobel X	1708937	132
Sobel Y	1562501	120
Sobel final	509303	39
Total	3780741	292

Zürcher Fachhochschule

14

Lets presume the use of a standard hardware, some processor running at 50MHZ with internal cache and external RAM. Remember the camera delivers 25 frames/second so there are

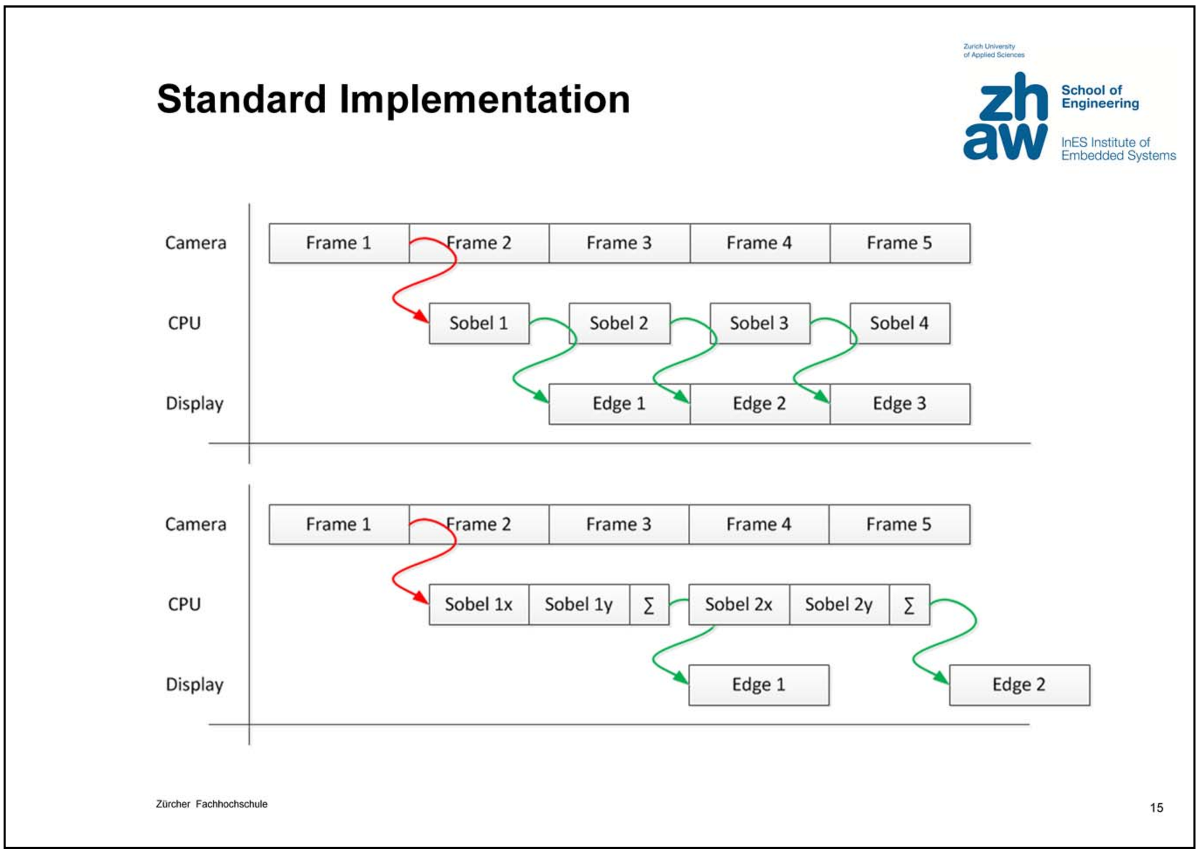
50E6 / 25 = 2E6 cpu cycles /frame available or, in other terms,

154 cpu cycles per pixel.

And that is assuming the hardware does nothing else.

Lets estimate the effort needed to perform the Sobel algorithm: X and Y directions require 4 multiplications each, a couple of additions and the final operation requires 2 absolutes as well as an addition and a subtraction and a set. So – ca 30 cpu cycles per pixel – no problem, n’est pas? – Wrong.

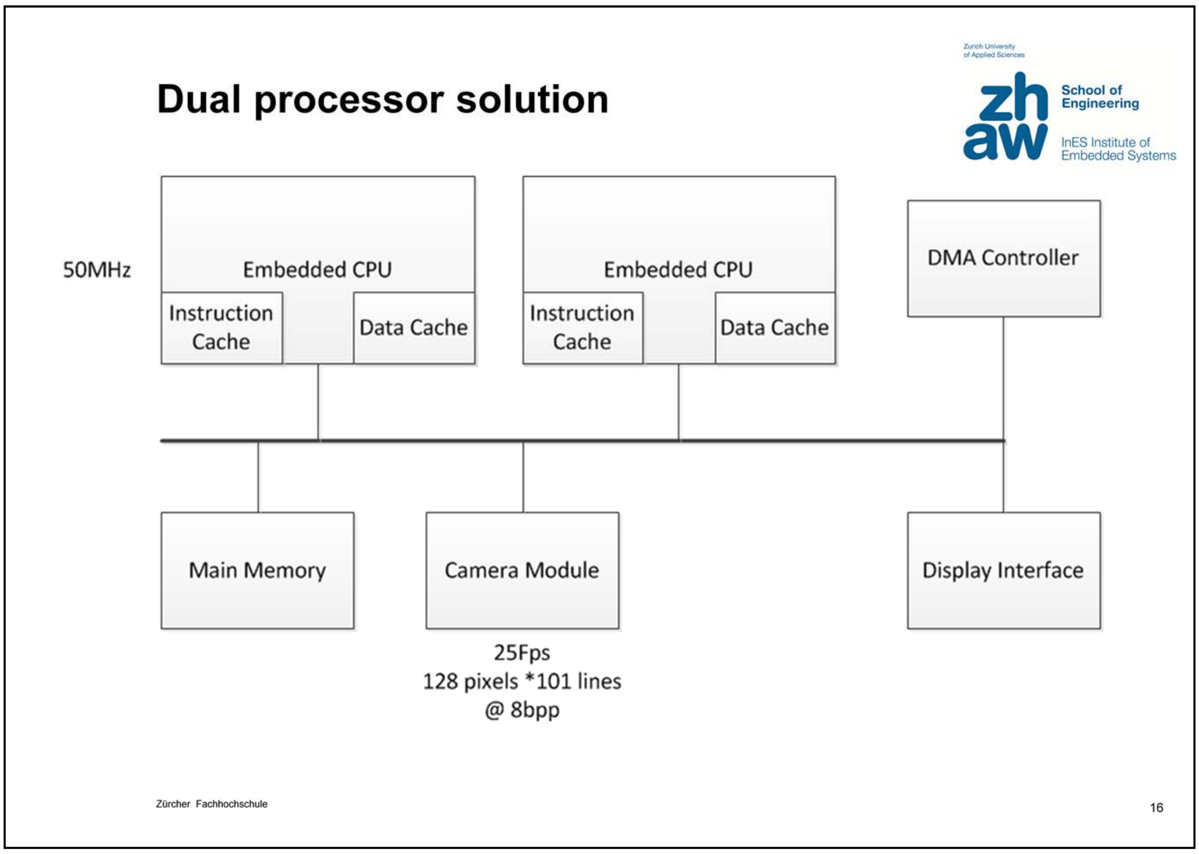
A basic C program performing the Sobel algorithm gives us the data as shown above: in other words we have an issue and we haven’t even discussed the hardware yet. – Lets take a look at the implementation and its timing.



Standard Implementation

Well then: in an ideal world our timing would look something like the top chart, But unfortunately it doesn't, it looks more like the bottom timing chart where measurement results say there application needs roughly twice the amount of time the CPU has at its disposal.

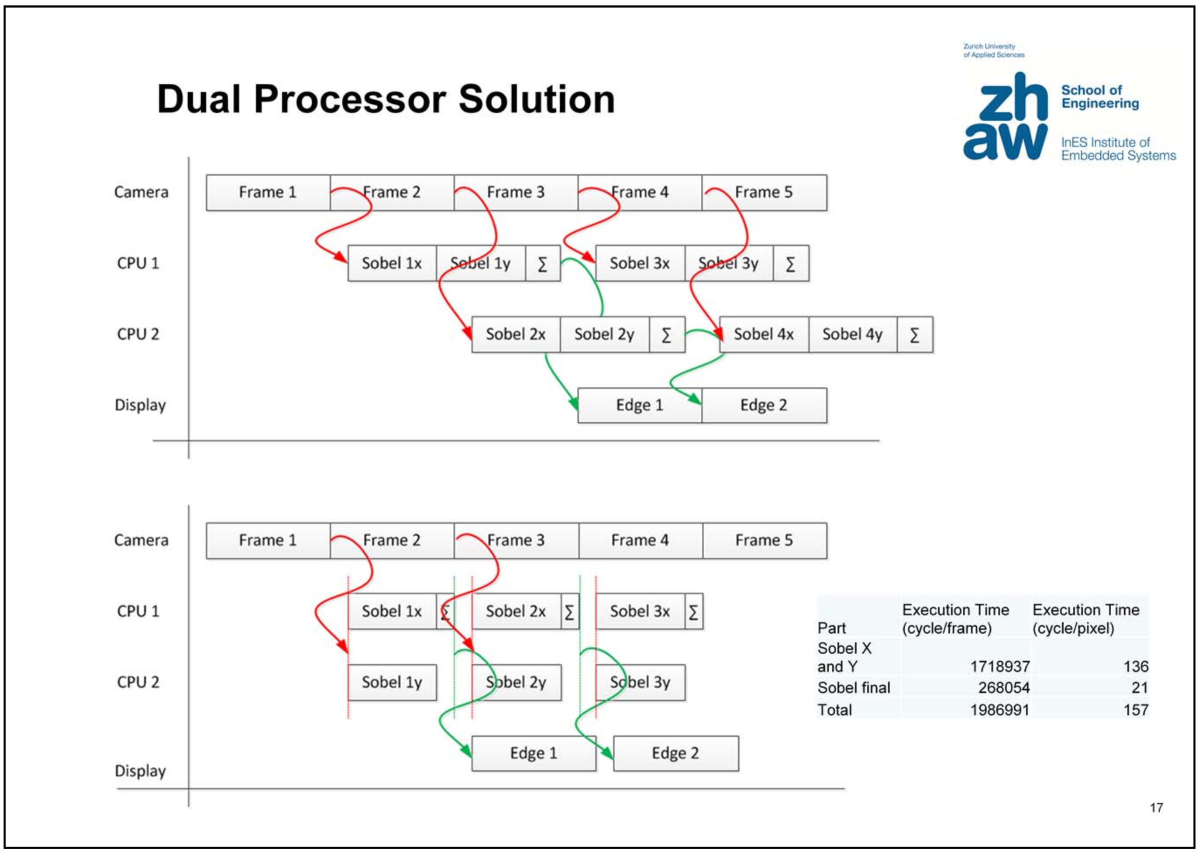
The question becomes then – what to do. If you assume that the processor can't be jacked up-to double the speed then you either need a new processor or add another processor.



Dual processor solution

By picking a second processor it is, in theory, possible just to double the processing power and solve whatever issue one might have – as I said, in theory.

Dual Processor Solution



Part	Execution Time (cycle/frame)	Execution Time (cycle/pixel)
Sobel X and Y	1718937	136
Sobel final	268054	21
Total	1986991	157

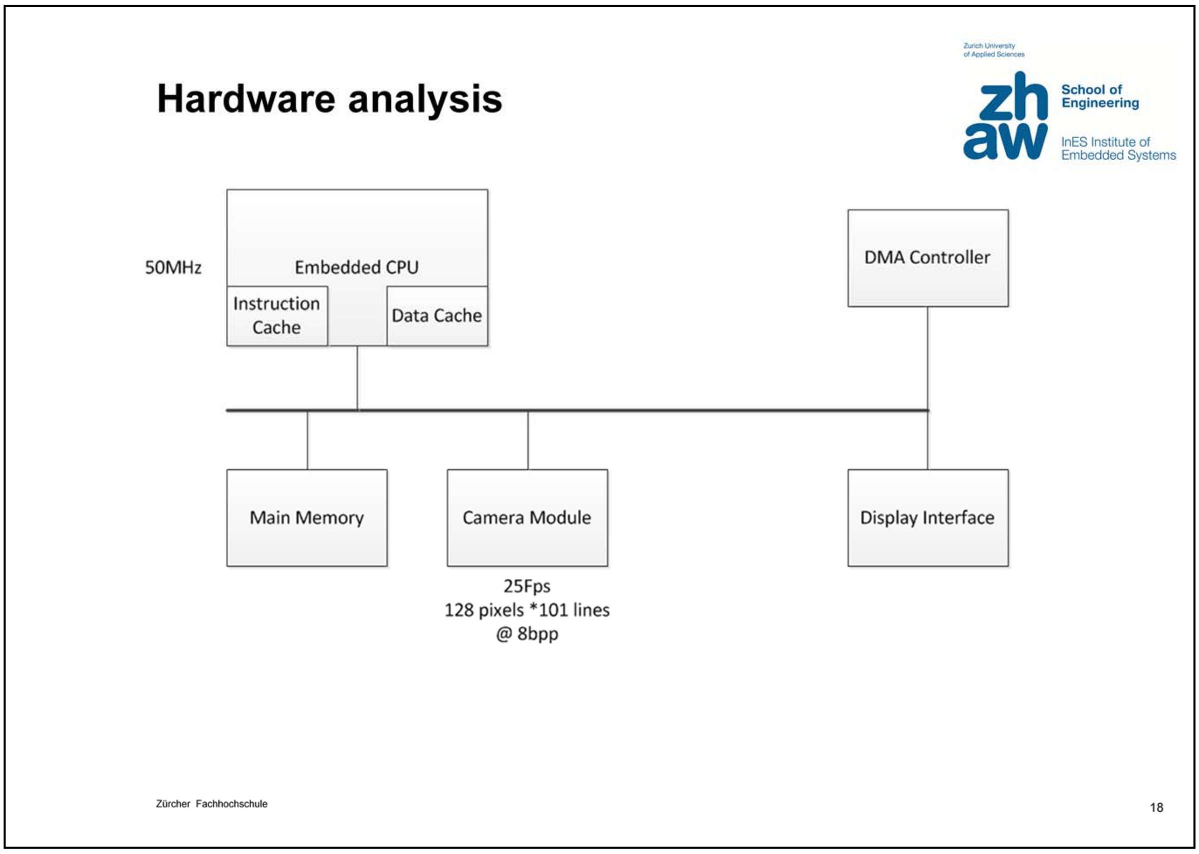
17

In the first potential solution is to assign each frame a processor – this allows us to ping pong between the two cores. (discuss the resources required for such a solution)

A second possibility would be to split up X and Y processing between the two cores and have one core do the final activities before allowing the display unit to display the results.

What we have done in an ad-hoc fashion is actually a formal design technique. We have partitioned the solution into HW and SW components here is achieved here is **mapping**– the distribution of a potential solution over a hardware/software set. What we have not achieved is to do this via standard hardware – in other words from a standard architecture that might have been used for the solution one now needs to develop a customised hardware in order to solve the problem.

Lets take a look at some of the HW issues involved



Hardware analysis

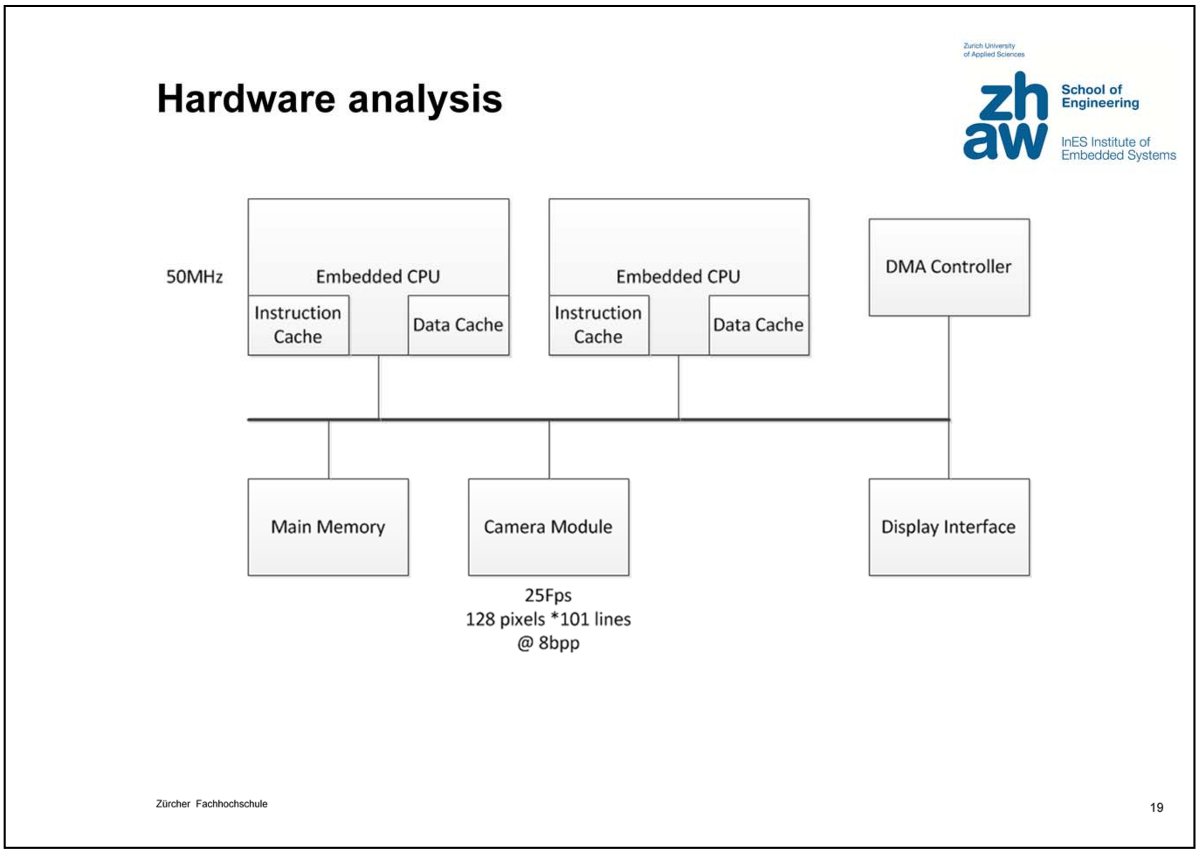
Assuming the camera module generates an interrupt when the picture is ready to be collected. There will be a latency between this interrupt and the actual beginning of the transfer. The magnitude of the latency is dependent on factors as whether the DMA unit can be programmed to react to an interrupt, hence the latency of the interrupt signal triggering the transfer or whether the processor must trigger an ISR which then programs the DMA unit. We call this latency L_{i1}

The DMA unit, before starting transfer, must request the bus from the default bus master, the CPU. The CPU grants the bus within a latency L_{BG}.

The transfer may begin and the data needs a certain number of clock cycles to be transferred from camera to main memory, plus generally a bit of overhead (address setup times and the like). Lets call this L_{BT}.

When the transfer is complete then the DMA unit will trigger an interrupt to the CPU which will call an ISR and determine the source of the interrupt. It will then trigger the Sobel task to deal with the image. Whether this trigger is achieved via a OS or a call-back is architecture dependent. Lets call these two latencies L_{tf} and L_{sp}.

Now the calculation of the Sobel algorithm can begin and finishes with latency L_s



Hardware analysis

Zurich University of Applied Sciences

zh aw School of Engineering InES Institute of Embedded Systems

Zürcher Fachhochschule 19

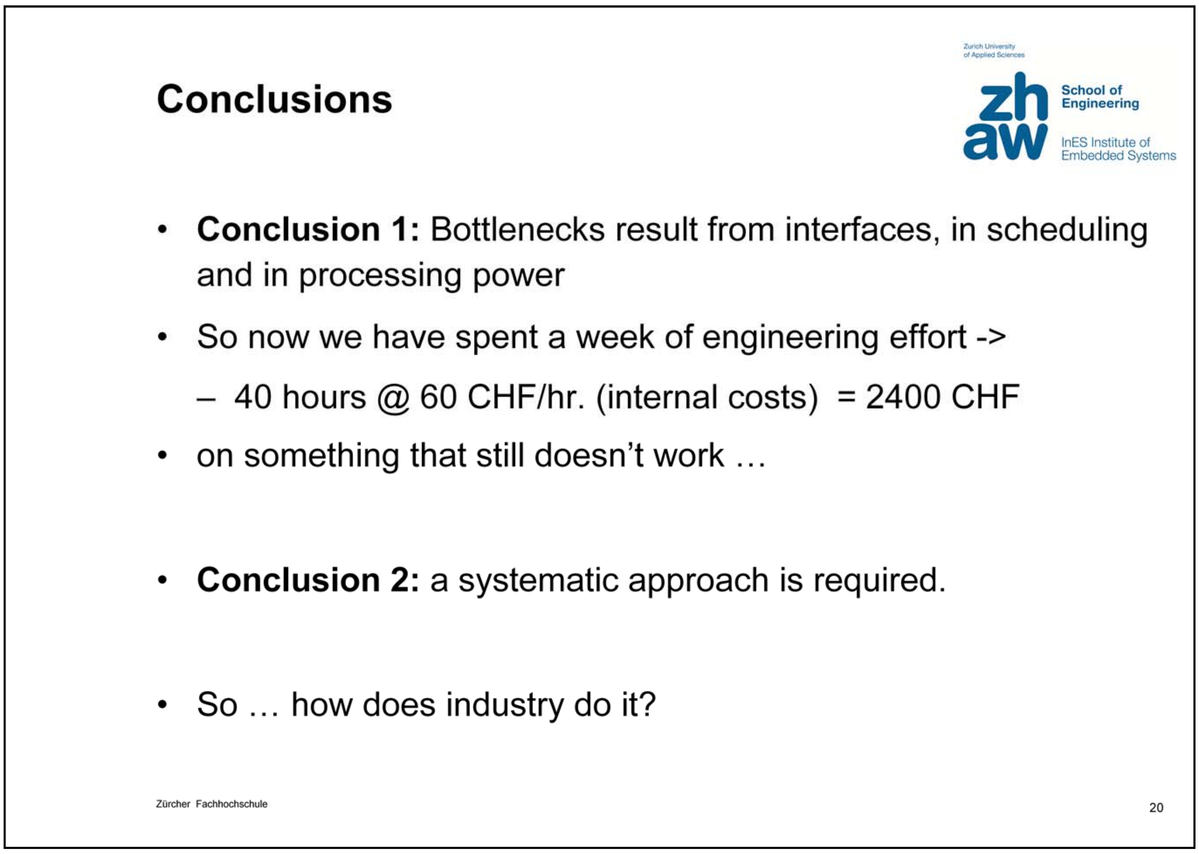
If we make the assumption that the instruction and data caches are turned off then during the calculation multiple accesses to main memory must be made – all imposing latency L_b – how large this latency is, is dependent on how the algorithm is written. L_b is actually a component of L_s, in other words $L_s = L_b + L_a$ (algorithm) where we would expect $L_b << L_a$. Finally we must trigger the DMA transfer of data to the display module (L_{t2}), the DMA unit must get the bus (L_{bg}) and transfer the data (L_{bt2}).

So a first expression for the latencies involved in getting a solution, resolves, by simple inspection, to:

$$L_{i1} + L_{bg} + L_{bt1} + L_{tf} + L_{sp} + L_s + L_{t2} + L_{bg} + L_{bt2}$$

So far we have been discussing L_s which can be imagined to be larger by far than the other latencies. But distributing the algorithm across two processors means designing a second processor into the hardware and this not only comes at a monetary cost it also affects the latencies.

In hardware – if we assume a standard 50MHz controller we are generally using a controller with an integrated DMA unit. Unfortunately these do not necessarily give up the bus to external entities, so either a new processor is chosen or the bus system must be redesigned. Then there is the issue of an external bus arbiter than must now arbitrate between three bus masters, the DMA controller and two microcontrollers.



Conclusions

Zurich University of Applied Sciences
School of Engineering
InES Institute of Embedded Systems

- **Conclusion 1:** Bottlenecks result from interfaces, in scheduling and in processing power
- So now we have spent a week of engineering effort ->
  - 40 hours @ 60 CHF/hr. (internal costs) = 2400 CHF
- on something that still doesn't work …

- **Conclusion 2:** a systematic approach is required.

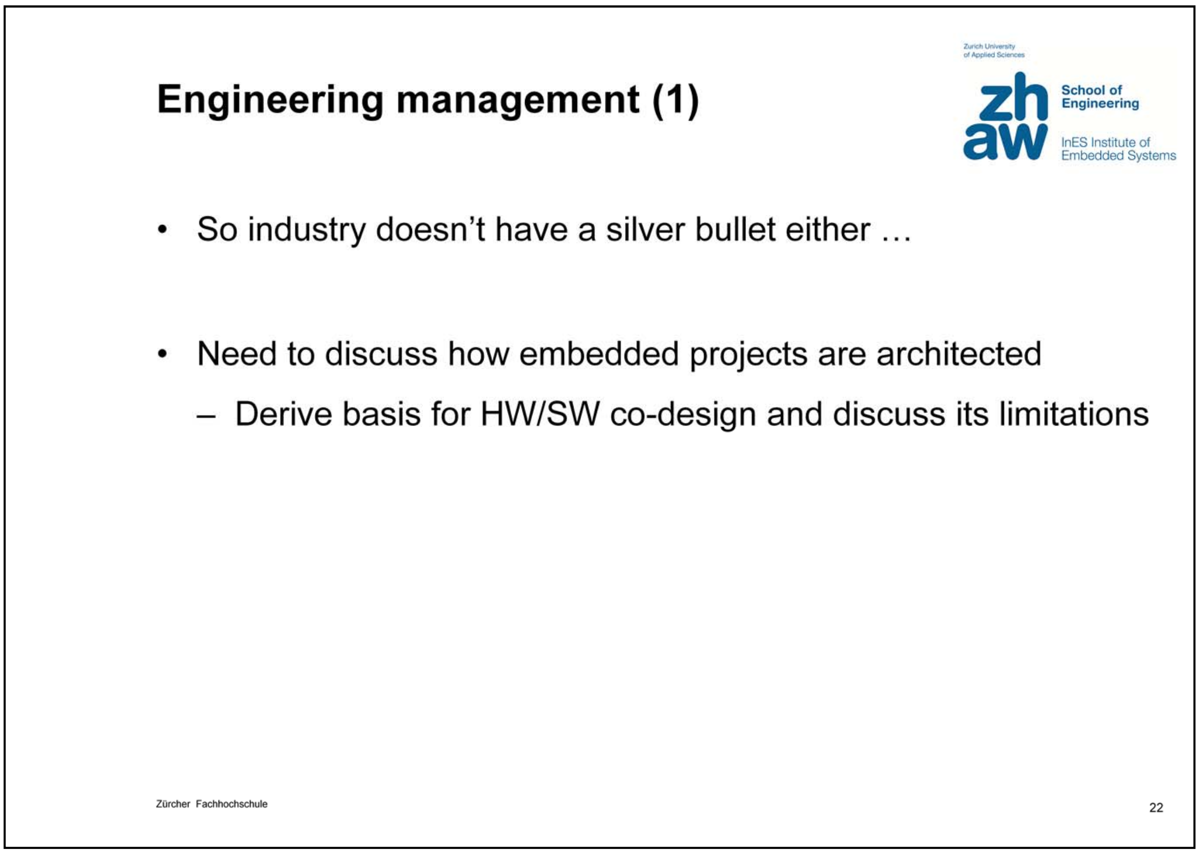
- So … how does industry do it?

Zürcher Fachhochschule 20

Context of Embedded Engineering

Zurich University of Applied Sciences
School of Engineering
InES Institute of Embedded Systems

- Notes on embedded.com survey



Engineering management (1)

Zurich University of Applied Sciences

zh aw School of Engineering

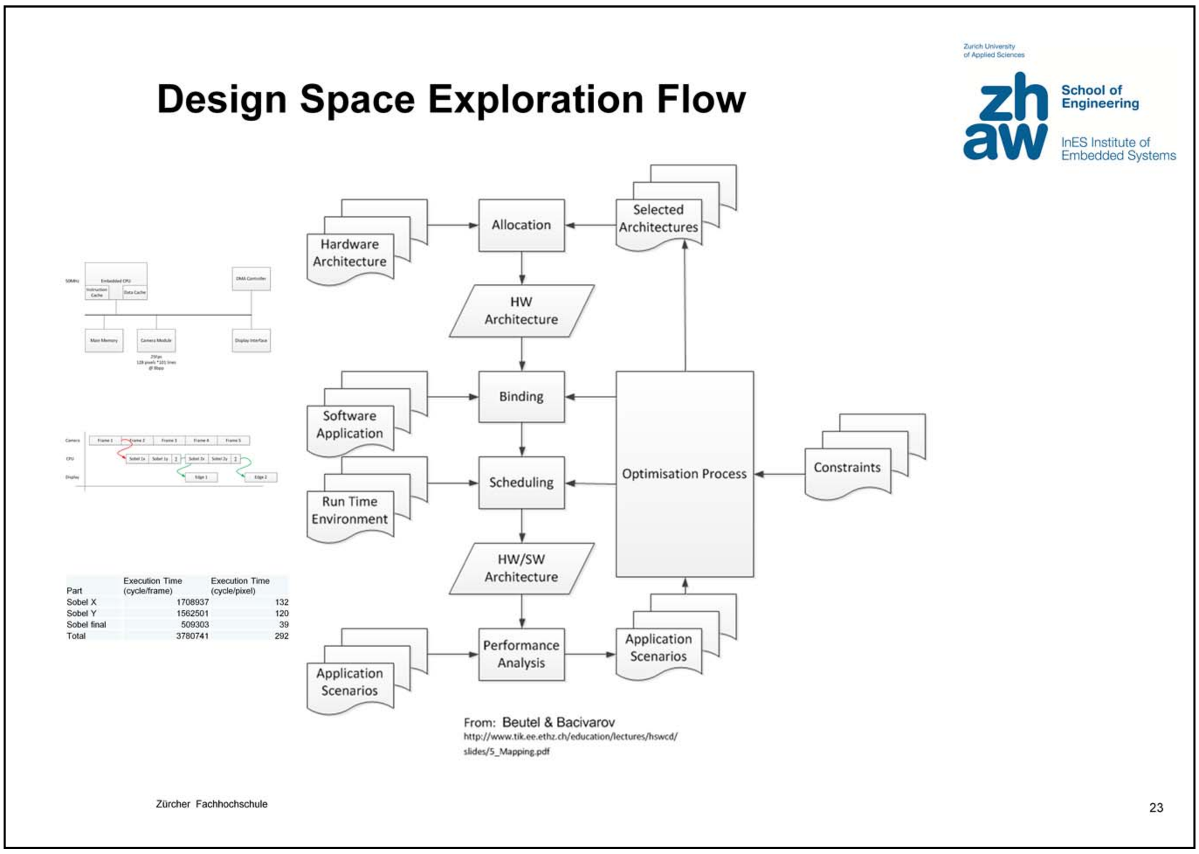
InES Institute of Embedded Systems

- So industry doesn't have a silver bullet either …
- Need to discuss how embedded projects are architected
  - Derive basis for HW/SW co-design and discuss its limitations

Zürcher Fachhochschule

22

Why: Obviously – if industry takes a fragmented view of decision making, those of you with a structured approach are in a clear advantage



Design Space Exploration Flow

Zurich University of Applied Sciences

zh aw School of Engineering

InES Institute of Embedded Systems

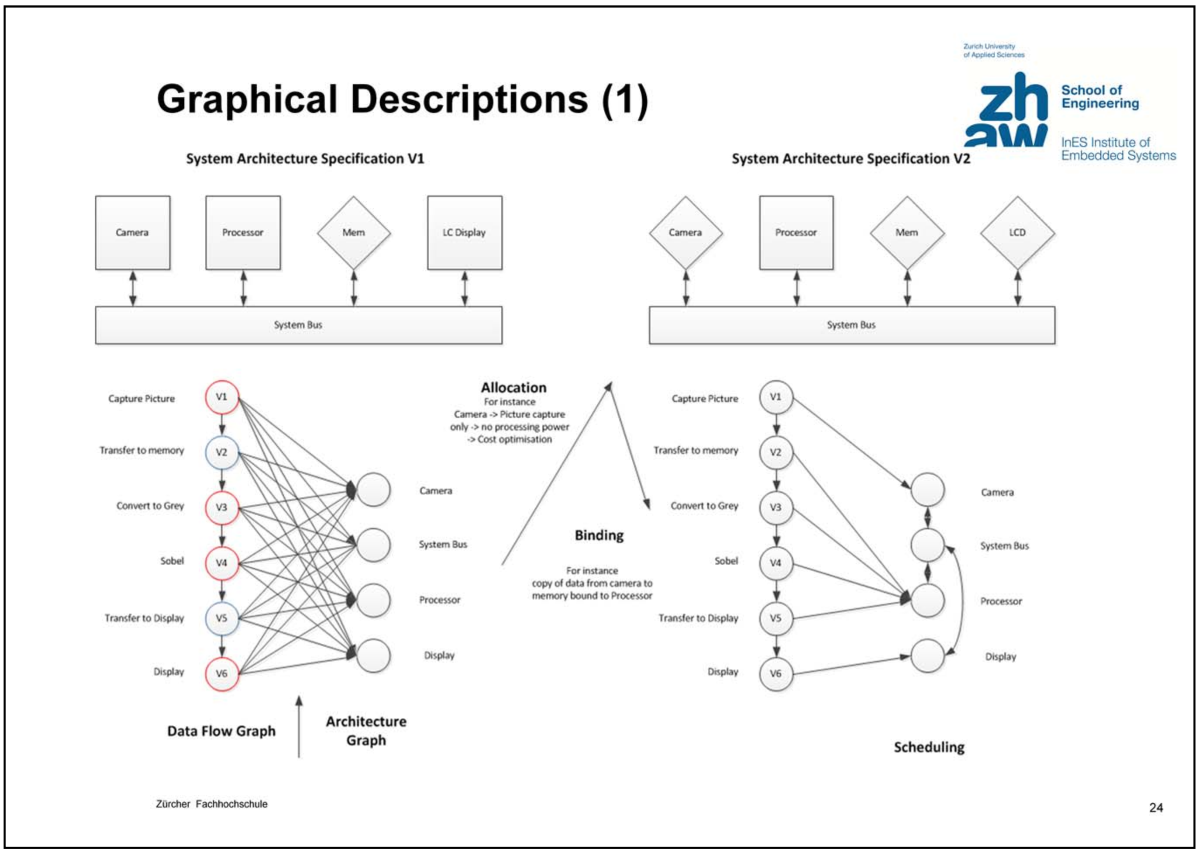
Part	Execution Time (cycle/frame)	Execution Time (cycle/pixel)
Sobel X	1708937	132
Sobel Y	1562501	120
Sobel final	509303	39
Total	3780741	292

From: Beutel & Bacivarov
http://www.tik.ee.ethz.ch/education/lectures/hswcd/slides/5_Mapping.pdf

Zürcher Fachhochschule

23

From a methodological point of view what we have done is we have selected components, in the first instance a standard hardware architecture – a process known as **allocation**. We have **bound** functions onto various components of this hardware, these two together form part of a process called **partitioning**. We have then determined the **execution order (scheduling)**, this together with the binding process is called **mapping**. With this HW/SW architecture we have then performed **performance analysis** and found our first iteration to be wanting. Assuming an extra controller and a re-binding/and scheduling we are engaging in a **multivariate optimisation** activity – cost (monetary) versus latencies and, potentially, power



Graphical Descriptions (1)

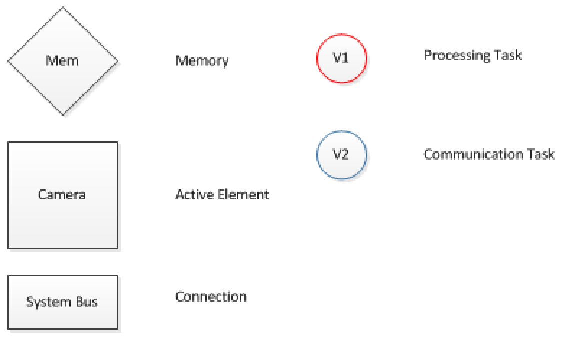
Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems
System Architecture Specification V1
Camera
Processor
Mem
LC Display
System Bus
Capture Picture
Transfer to memory
Convert to Grey
Sobel
Transfer to Display
Display
V1
V2
V3
V4
V5
V6
Camera
System Bus
Processor
Display
Data Flow Graph
Architecture Graph
Allocation
For instance
Camera -> Picture capture only -> no processing power -> Cost optimisation
Binding
For instance
copy of data from camera to memory bound to Processor
System Architecture Specification V2
Camera
Processor
Mem
LCD
System Bus
Scheduling

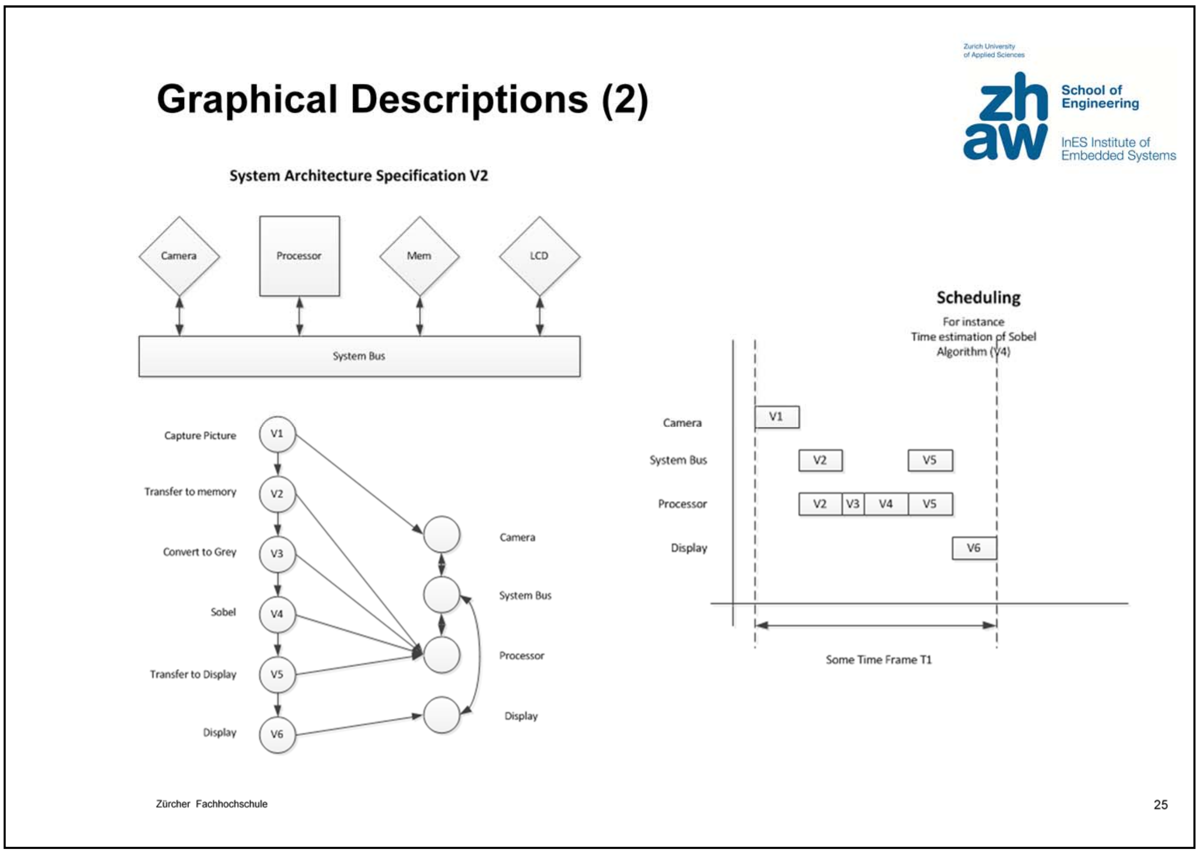
Zürcher Fachhochschule 24

The system architecture shows three active units and a memory unit. The data flow graph shows the flow of raw data to the final product. The arrows show read/write relationship on the system bus and the implication is that every active device can communicate to/from the memory

The architecture graph shows how the data flows from camera to LCD in discrete processing steps. As a starting position each process is bound to each element of the architecture graph. Then they are allocated to specific active units and we get the system architecture shown right above. Note how the camera and LCD are now classed as memory devices. The binding adopted shows how virtually all activities are delegated to the processor. It also better illustrates how much transfer goes on via the system bus.

Key:



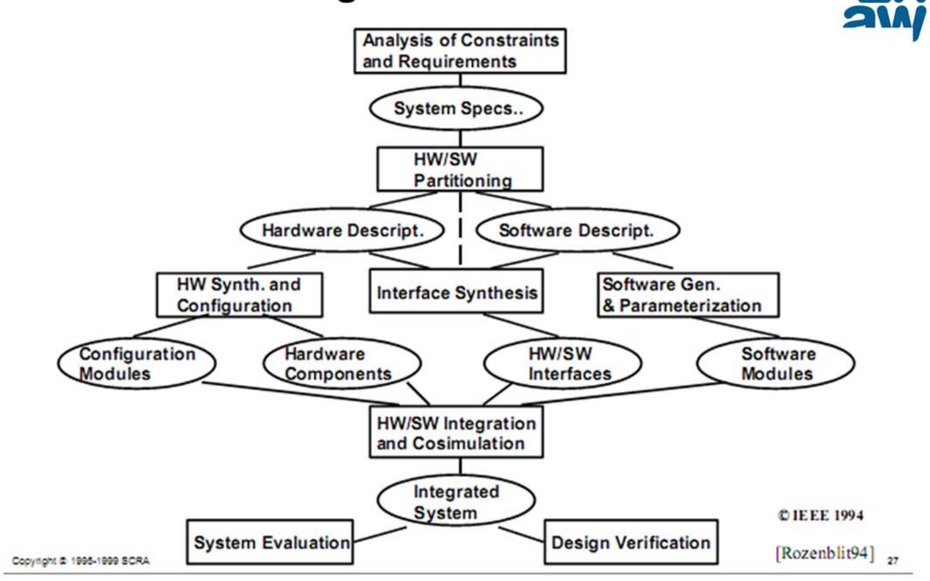


Graphical Descriptions (2)

Zürcher Fachhochschule 25

Largely a repeat of the previous slide on the right hand side the system schedule is shown

HW SW Co-Design



It is practically rare to have a hardware-only system – early analog systems such as the ones that sent people to the moon have largely been replaced by digital systems with appropriate state machine, software and other automatons performing tasks.

HW-SW co-design is an attempt to methodologically design a complex (generally) embedded system that optimises some cost function. The methodology is shown above.

Here – as is consistant with our introduction to the V-model, HW-SW co-design starts with the architecture – after definition of the system specifications. The partitioning divides the design space into HW and SW units and the definition of interfaces between these. Then comes the synthesis of the various parts and then eventually the joining together (HW/SW Integration) before test and verification.

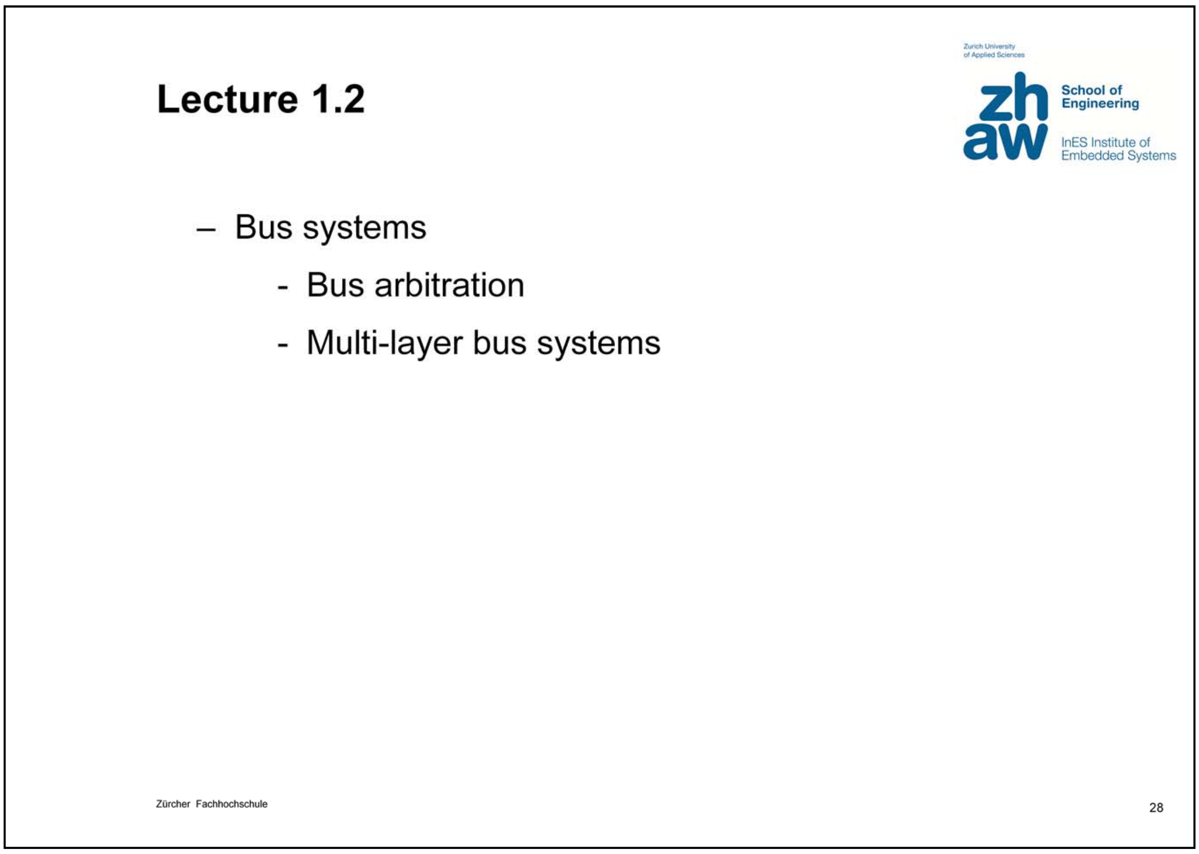
A large part of research in HW-SW co-design centres on the automated finding of a minimum for the cost function and is therefore very tool orientated. In this module we opt for a different, but equally valid, approach and that is to do the job in SW and then sucessively find an optimal solution in HW and SW.

Cost Function

- Tool based HW-SW Co-Design generally revolve around the optimisation of some cost function:
  - Cost (C), Power (P), Latency (L)

$$f(C, L, P) = k_c * h_c(C, C_{\max}) * k_p * h_p(P, P_{\max}) + k_L * h_L(L, L_{max})$$

  - k is a normalisation constant
  - h(n, n) is the degree of violation of max constraints



Lecture 1.2

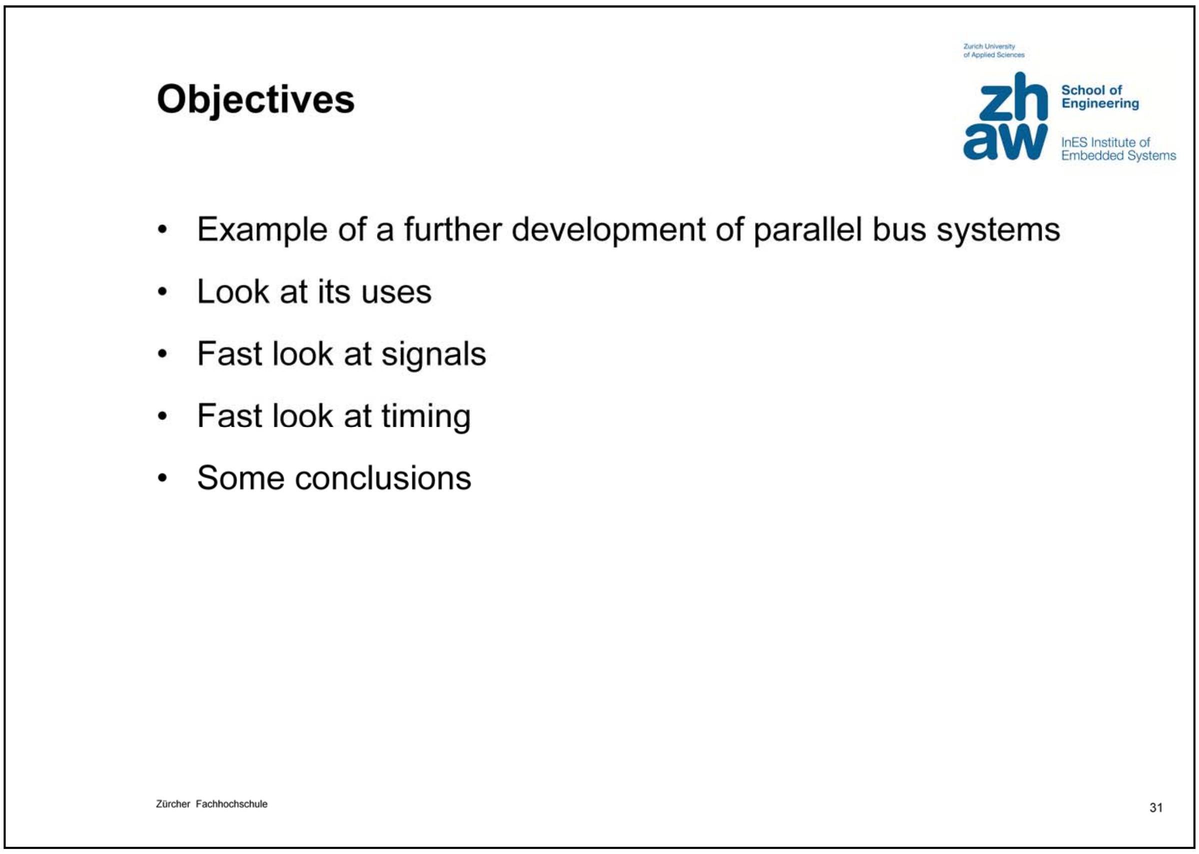
- Bus systems
  - Bus arbitration
  - Multi-layer bus systems

Literature

- AMBA System Specification Rev 2. 1999.
- AMBA Multi-Layer AHB Overview Rev B 2004
- AMBA AXI Specification V1.0 Rev B 2004

Learning Aims

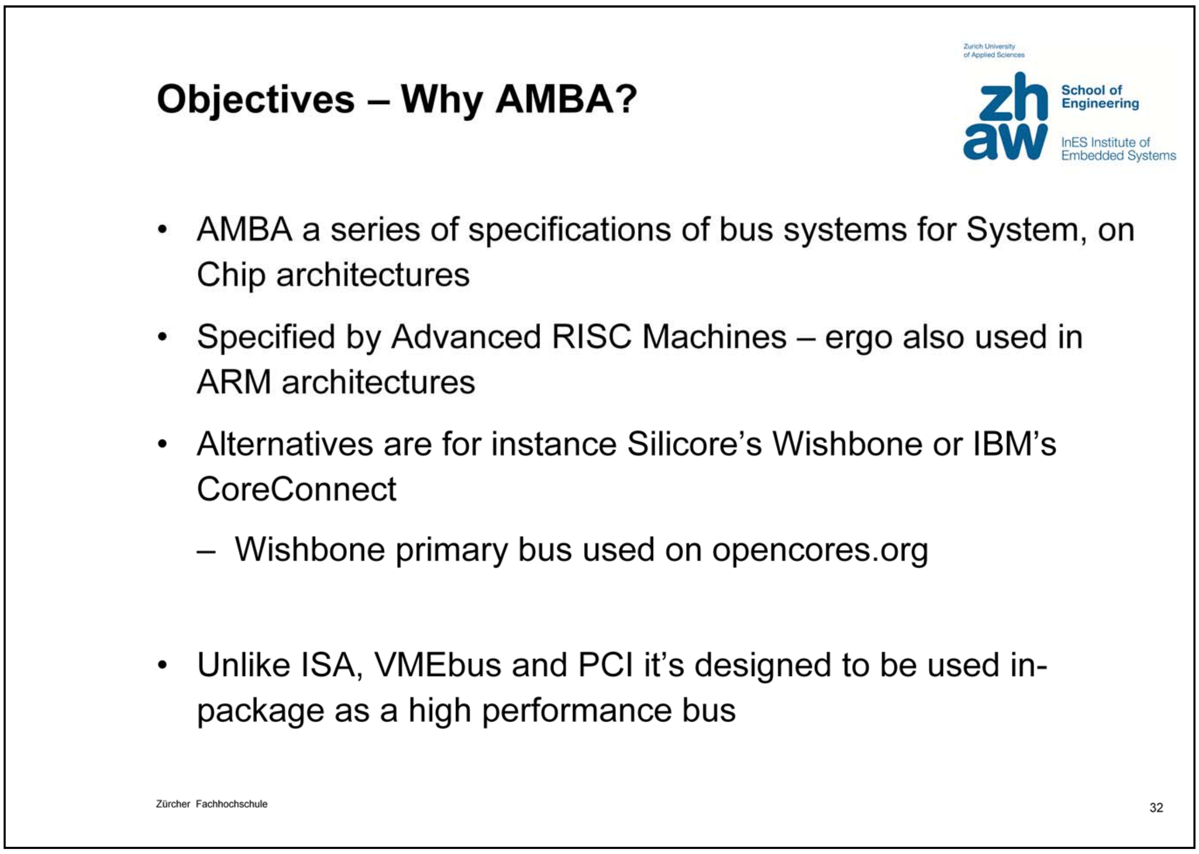
- The student will understand and be able to understand the general functionality of the bus systems defined in the various AMBA specifications.
- The student will be able to explain the effects of bus arbitration on task timing
- The student will be able to



Objectives

Zurich University of Applied Sciences
School of Engineering
InES Institute of Embedded Systems

- Example of a further development of parallel bus systems
- Look at its uses
- Fast look at signals
- Fast look at timing
- Some conclusions



Objectives – Why AMBA?

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems

- AMBA a series of specifications of bus systems for System, on Chip architectures
- Specified by Advanced RISC Machines – ergo also used in ARM architectures
- Alternatives are for instance Silicore's Wishbone or IBM's CoreConnect
  – Wishbone primary bus used on opencores.org
- Unlike ISA, VMEbus and PCI it's designed to be used in-package as a high performance bus

Zürcher Fachhochschule 32

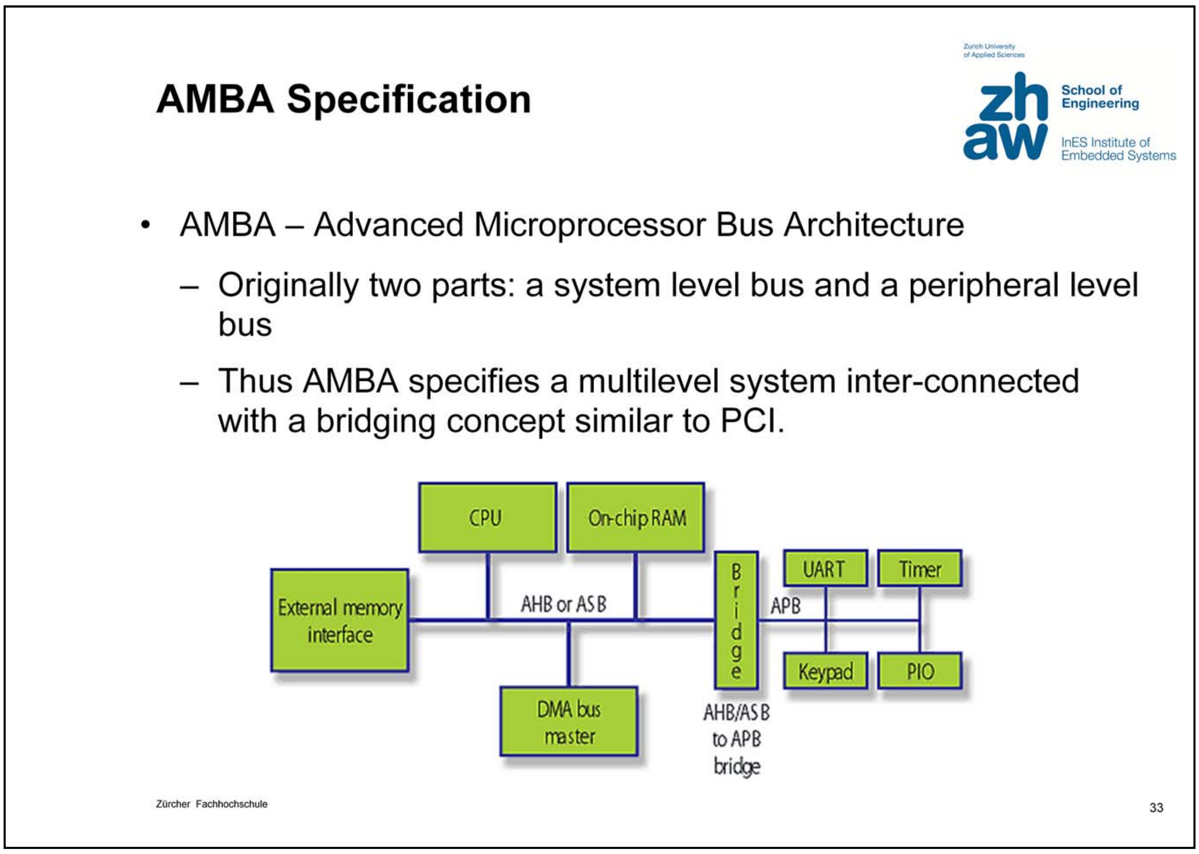
History: Advanced RISC Machines started off as the design house of Acorn computers. In the early 80's the market for home computers was completely open and several national companies, Sinclair, Atari, Acorn, Olivetti, Wang, Siemens, made their own PC's. Some with astounding success.

Success came to Acorn with the adoption of their machines in a BBC (British Broadcasting Corporation) home-computer course. The initial machines were Rockwell 6500 based machines. Whilst the Acorn was a great machine it soon became obvious that 8-bit 8MHz machines weren't going to rule the world and that there were no suitable 16 or 32- bit machines on the market. Being British, Acorn decided to build their own.

This they did with remarkable success. The first prototype was manufactured at VLSI Technology Inc. in the US and was delivered in 1985. However at that time Apple had decided to use the 68000 and the rest of the world was consolidating behind the IBM "standard". Acorn went into financial problems and was bought by Olivetti in 1985. By 1990 the ARM architecture was fairly established and very interesting due to its low-power capabilities (ARM-2). Nevertheless Acorn couldn't afford to keep an in-house processor development team and ARM Ltd was founded with Apple, Acorn and VLSI Technology as founding partners. The major difference was that ARM was not to produce chips itself but license its designs to chip makers who would pay it royalties.

The ARM6 (actually the ARM4 but a change in numbering took place) was used in Apples Newton. Sharp was the first Japanese licensee.

Purpose AMBA: Since ARM doesn't produce discrete microcontrollers (Luminary Micro, now Texas, was set up by ARM guys wanting to commercialise the Cortex architecture) their focus is on developing IP meaning the interface to their IP must be capable of being used in a multitude of different situations,, as a silicon interconnect bus, as a pin-out bus on a uC package and as a routable bus in a GA or FPGA hardware. However the bus is independent of the ARM hardware.



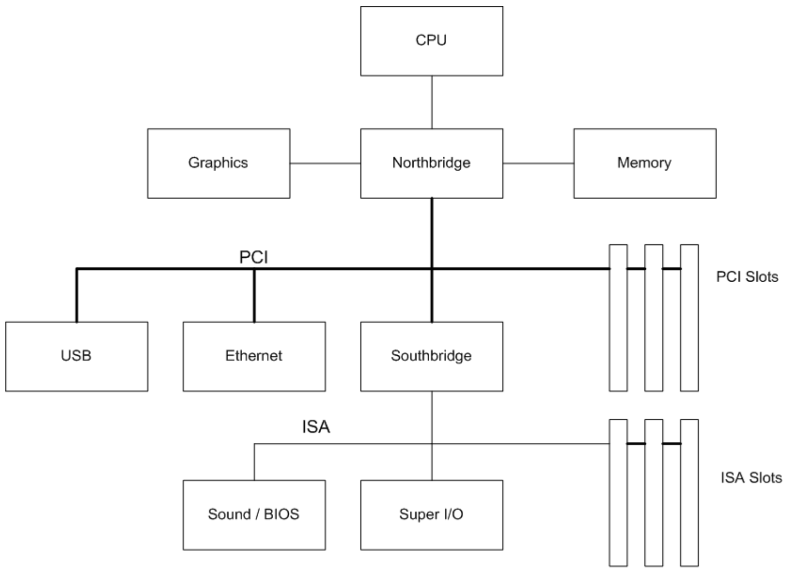
AMBA Specification

- AMBA – Advanced Microprocessor Bus Architecture
  - Originally two parts: a system level bus and a peripheral level bus
  - Thus AMBA specifies a multilevel system inter-connected with a bridging concept similar to PCI.

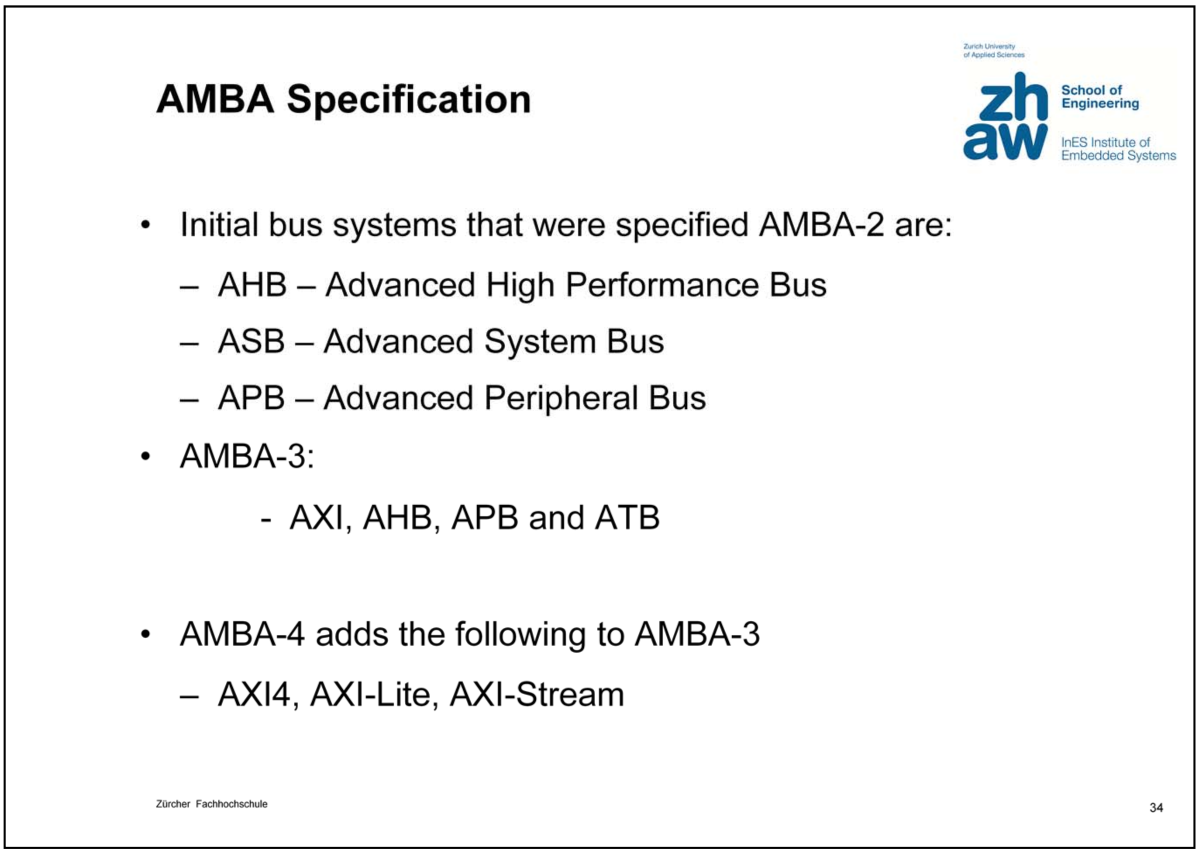
CPU
On-chip RAM
External memory interface
AHB or ASB
DMA bus master
Bridge
AHB/ASB to APB bridge
APB
UART
Timer
Keypad
PIO

Zürcher Fachhochschule 33

Multilevel System: See the diagram of PC where the processor is separated from the PCI by the Northbridge and ISA from PCI by the Southbridge? Well this is in essence what AMBA specifies. The AHB/ASB for system level components and the APB for peripheral components, connected together by a bridge, but actually bearing no other relation to each other.



PC Architecture with PCI (ca. 2001)



AMBA Specification

Zurich University of Applied Sciences

zh aw School of Engineering

InES Institute of Embedded Systems

- Initial bus systems that were specified AMBA-2 are:
  - AHB – Advanced High Performance Bus
  - ASB – Advanced System Bus
  - APB – Advanced Peripheral Bus
- AMBA-3:
  - AXI, AHB, APB and ATB
- AMBA-4 adds the following to AMBA-3
  - AXI4, AXI-Lite, AXI-Stream

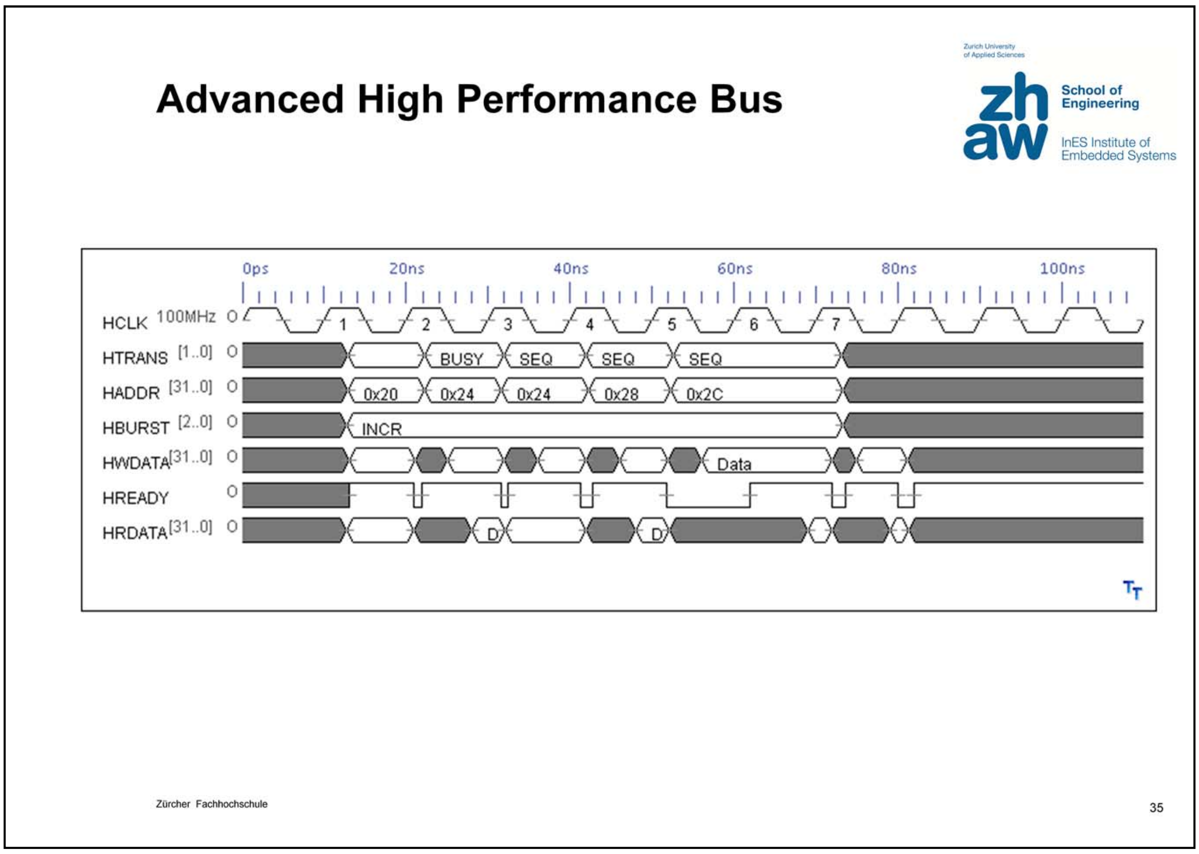
Zürcher Fachhochschule 34

Multilevel System: Remember the diagram of PC where the processor was separated from the PCI by the Northbridge and ISA from PCI by the Southbridge. Well this is in essence what AMBA specifies. Except there is a sort of Southbridge and no Northbridge, as explained below. The AHB/ASB is designed for system level components and the APB for peripheral components. The two are connected together by a bridge, but actually bear no other relation to each other.

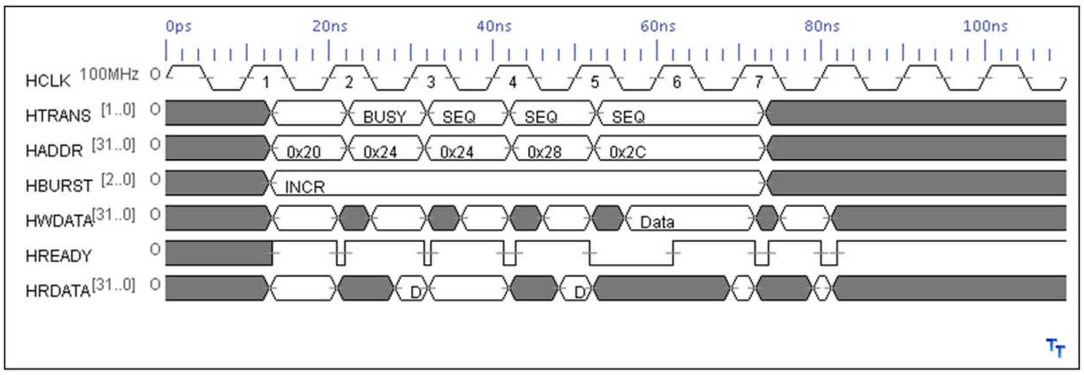
Efficient (ASB): Obviously ARM had to start somewhere so their first version was ASB. AHB isn't a superset of ASB, there are different signal groups. It is of course necessary to keep the faith with older specifications during a phase-out period and seems to have been written out of the AMBA-3 specifications whereas there is always need for a peripheral bus which is why the APB lives on.

High Performance (AHB): In our PCI block diagram the processor was connected to the system memory using the Northbridge. This is a fairly expensive option to take, usually too expensive for embedded and other cost-sensitive system, implying as it does a chipset. So the system bus in this architecture also means that the processor accesses the bus for both code and data. With a bridging solution to the lower and simpler, thus cheaper (implementation register numbers or number of gates), I/O bus APB something approaching an optimal architecture may be arrived at. AHB has a feature capability comparable to PCI.

Test Bus (ATB): I haven't gotten my hand on the specs yet but various Cortex processors offer this bus as a test interface. The importance of inclusion of test and trace busses in on-chip designs cannot be underestimated as the sheer complexity of designs makes verification exceedingly difficult and bus tapping can reduce debugging and long-term test effort considerably.



Advanced High Performance Bus



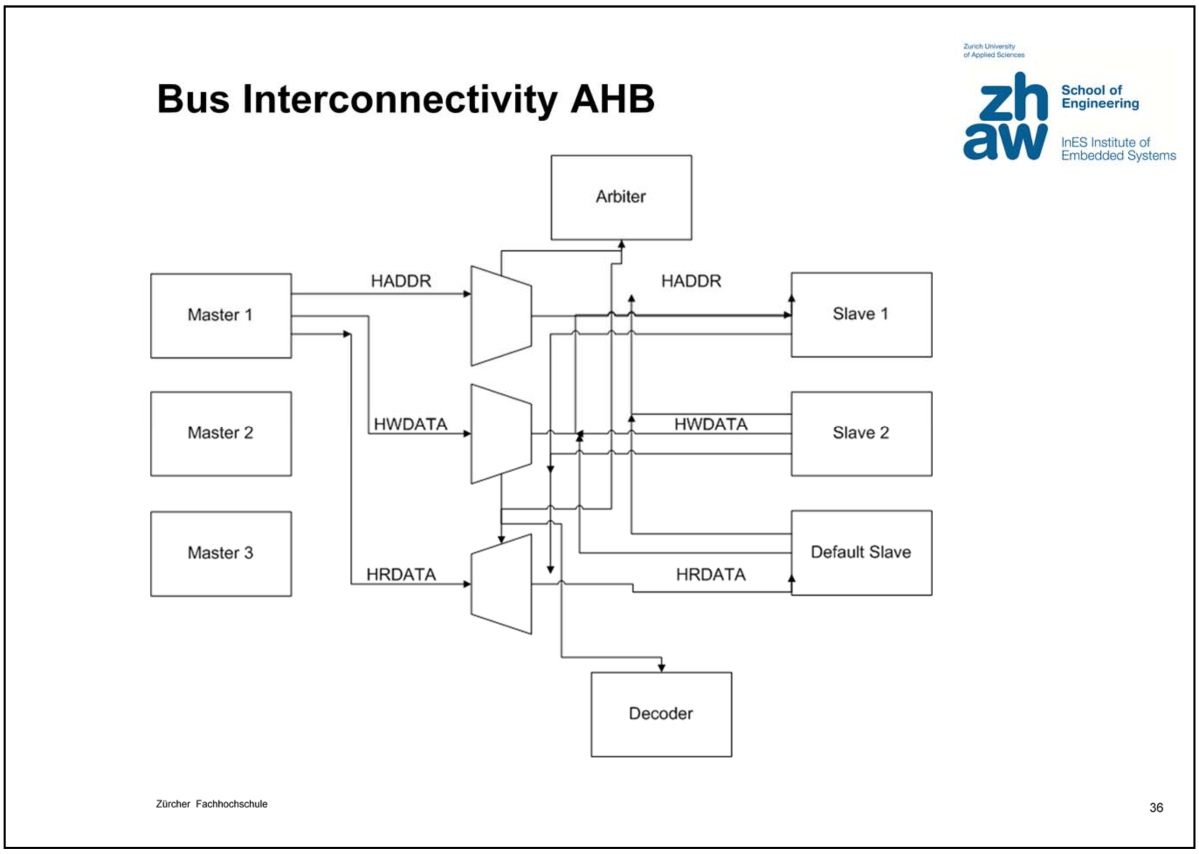
Zürcher Fachhochschule 35

Signalling: AHB signals are denoted by the prefix H. Signals are, with the exception of HRESETn active high. The rising edge of HCLOCK is the important one. Since the AHB is ***independent*** of clock *frequency*, all timing references are made to this edge as well. AHB boasts a 32-bit address bus although this is scalable to 64 and 128 bits without further notice or modification. HSELx is the slave-select signal which is simply a decode of the address bus. HTRANS indicates the type of **bus transfer**. These can be NONSEQUENTIAL, SEQUENTIAL, IDLE or BUSY. IDLE's are inserted by the **Bus Master** and are to be ignored by the **Bus Slave**. A master may insert BUSY into the middle of a cycle in order to tell the slave that the next transfer can not take place immediately. This transfer must be ignore by the slave. Burst transfers are denoted by a SEQUENTIAL although the first phase, as the address and data are unrelated to the previous transfer, is denoted by a NONSEQUENTIAL transfer type, as are single transfers.

HWRITE is high for a write transfer and low for a read transfer. HSIZE (not shown above) denotes the size of transfer (8-1024 bits). HBURST denotes whether the transfer is part of an undefined, 4,8, or 16 beat burst. Bursts may be incrementing or wrapping. BURSTS may not cross a 1kB address boundary. A wrapping transfer will wrap at a 16-byte boundary. HPROT[3:0] (not shown above) signals indicate whether this is an op-code fetch, data access, supervisor or user mode transfer.

HWDATA is the **write data bus**, 32 bit minimum is specified but is upwards scalable, as is the separate HREAD read data bus. This separation allows the bus to be used in environments were three-stating is not possible or desired (f.i. FPGA's). The bus is endian free. HREADY indicates that a transfer is finished, slaves need it as an input and an output. If the signal is driven low then a wait state is inserted HRESP[1:0] (not shown) indicates the success of a transfer, a Slave may respond with OKAY, ERROR, RETRY or SPLIT.

In order to ensure correct operation in systems where the address map is not filled a **default slave** should be implemented which defaults to being addressed when no legitimate slaves are addressed and issues ERROR signals when it is accessed.

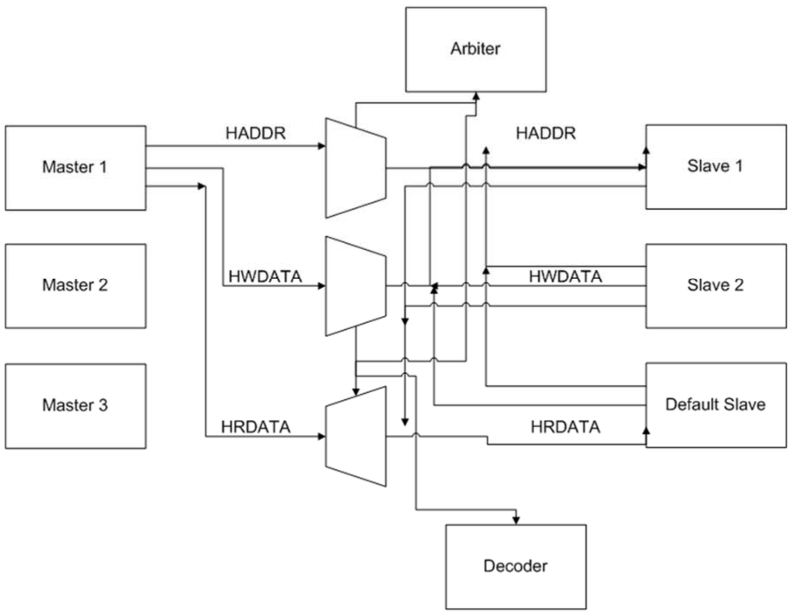


Bus Interconnectivity AHB

Zurich University of Applied Sciences

School of Engineering

InES Institute of Embedded Systems



Zürcher Fachhochschule 36

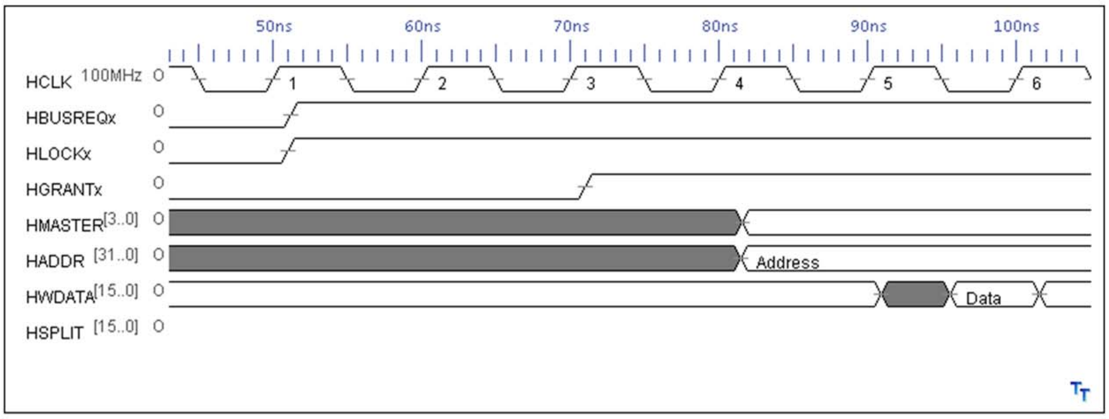
The diagram isn't really very good but the essential features are that whilst the bus is multi master the various masters/slaves are connected to each other via multiplexors. The arbiter will decide which master is allowed to connect to the slave side of the bus. The number of interconnects is fearsome but a direct result of the decision not to use three-state busses, as would be the case in an on-PCB bus.

Bus Arbitration AHB

Zurich University of Applied Sciences

zh aw School of Engineering

InES Institute of Embedded Systems



Zürcher Fachhochschule 37

In arbitration the potential master requests the bus of the arbiter by asserting the HBUSREQx line. There can be up-to 16 potential bus masters in a system. If the master intends to keep the bus for a series of accesses then it asserts the appropriate HLOCKx. If the bus is granted by the arbiter which decides the priority of the request, then the master may proceed with the multiple transfers. A burst transfer may be interrupted by the arbiter to grant a higher priority master the bus. Upon being given the bus the master asserts his priority on the HMASTER bus. The ownership of the data bus is delayed from that of the address bus (pipelining in effect).

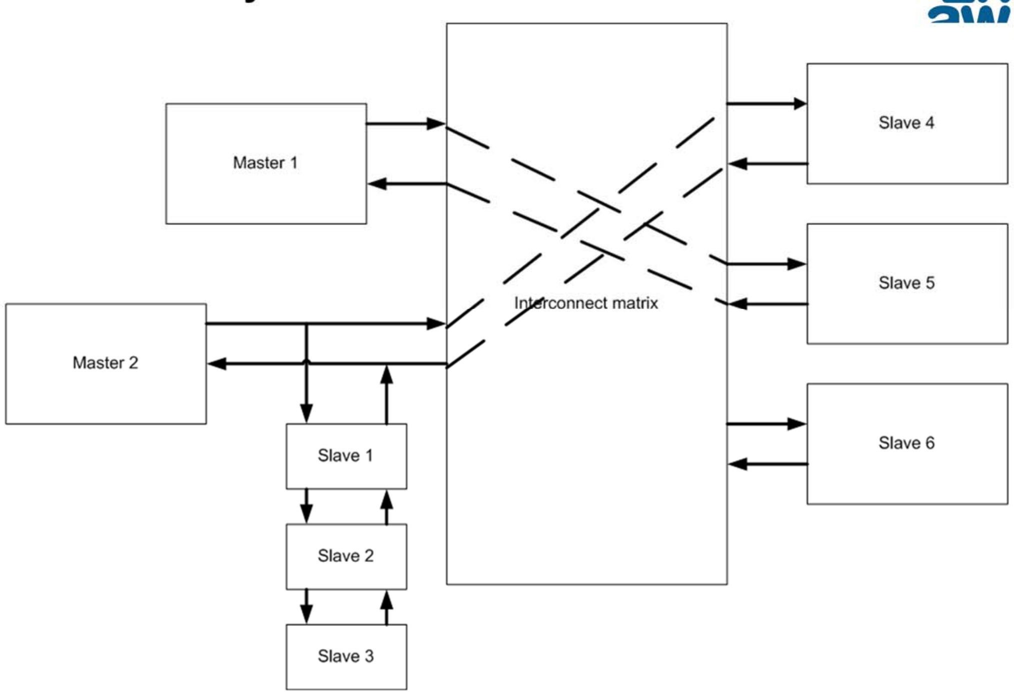
A default master is required when implementing an arbiter and must be nominated by the arbiter. This default master may only perform IDLE transfers when in default master mode.

Multi-Layer AMBA

- AMBA is not only multi-level but also multi-layer
  - Multi-level = system and peripheral bus
  - Multi-layer = parallel access between multiple masters and slaves
- Topology change not a protocol change
- Allows different variations of AHB
  - For instance AHB-Lite.

Multilayer AHB

Zurich University of Applied Sciences
School of Engineering
InES Institute of Embedded Systems

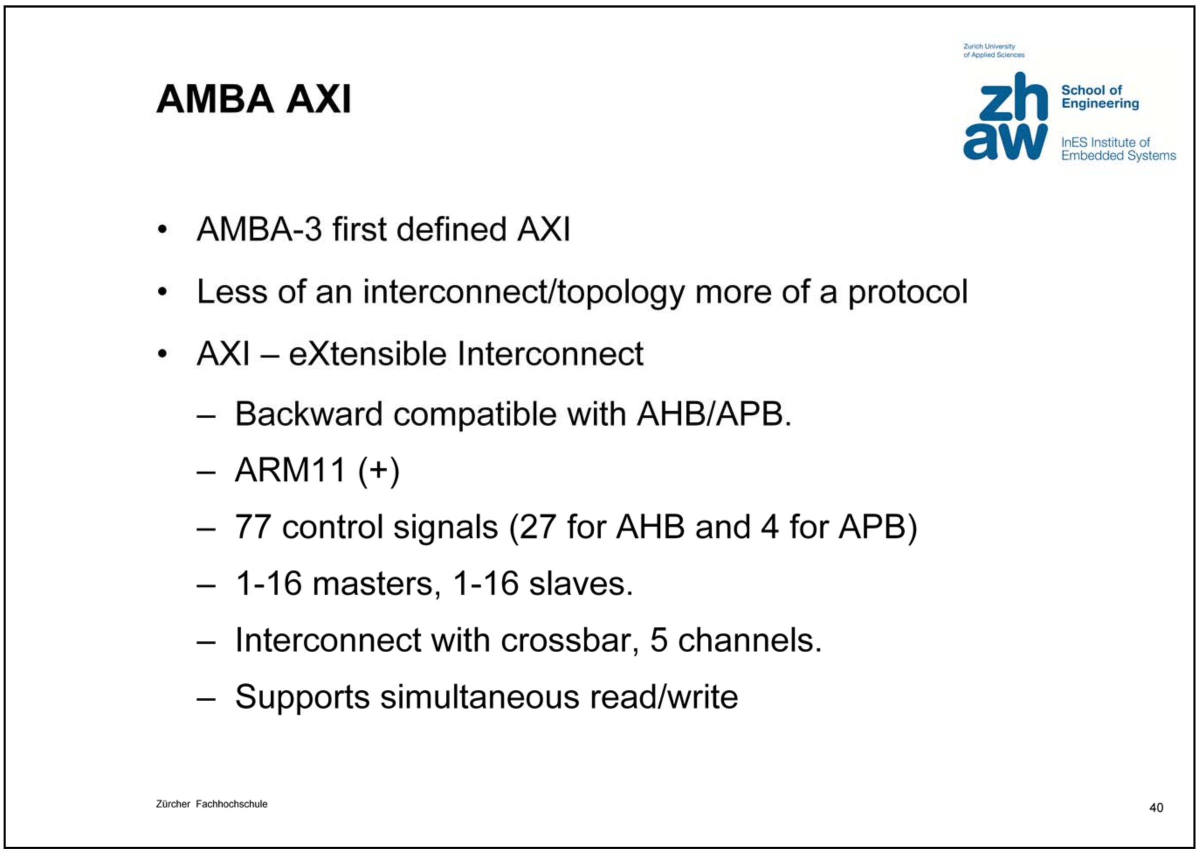


39

Interconnect: Essentially this topology change is an extension of the multiplexer system seen in AMBA-2 by a more complex interconnect stage. Based on the idea that masters rarely need to communicate directly with each other, and can’t in a system which doesn’t allow the use of tri-state bus, the interconnect stage solves potential bottleneck situations where one master must wait for the appropriate bus grant due to a second master hogging the bus. Obviously this situation might occur if two masters wish to access the same slave, in which case the second master is held up by the de-assertion of the HREADY signal but the potential bandwidth savings are obvious.

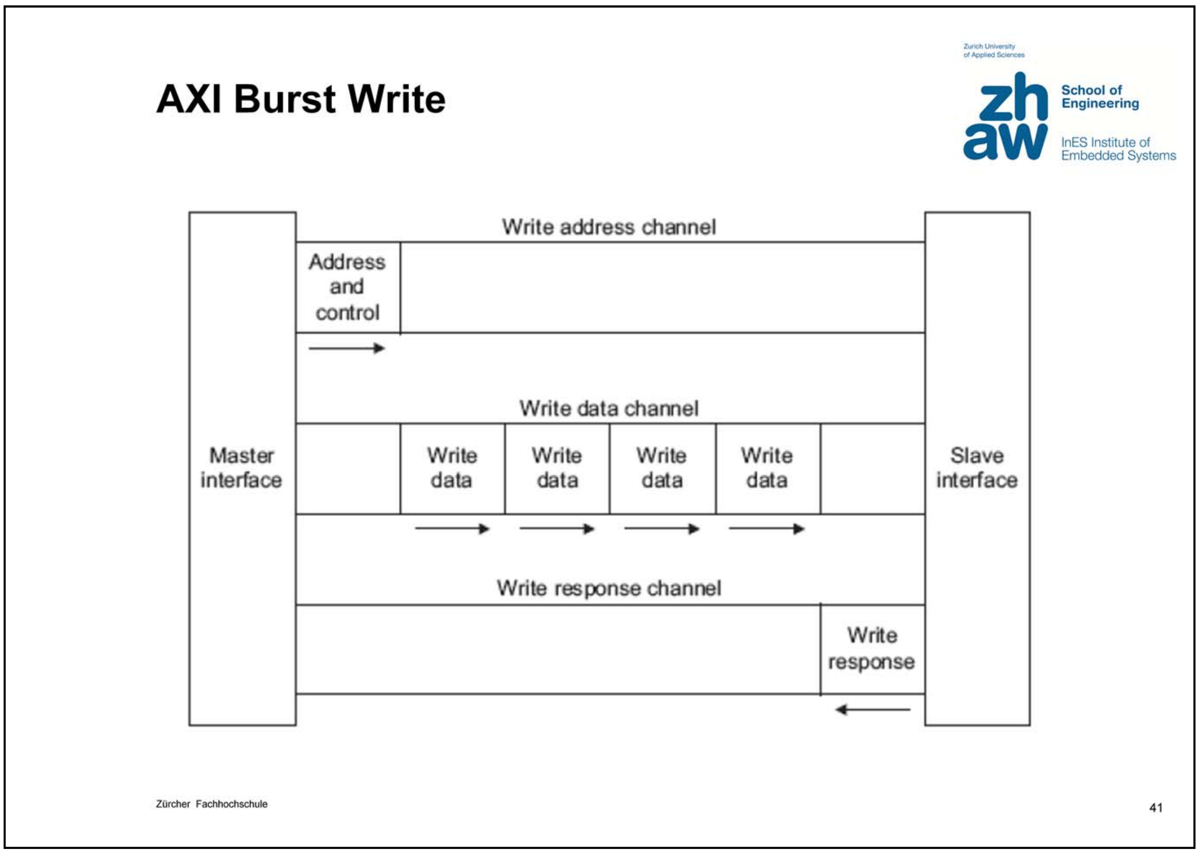
This kind of topology, and the use of the HREADY signal is very telling. In effect the AHB system does not need to support an arbiter, respective arbiter functionality is built into the interconnect matrix, leading to the AHB-Lite version of AHB – not supporting arbitration signals.

The general idea is that the bus is divided up into layers. The simplest form being that every master represents his own layer.



AMBA AXI

- AMBA-3 first defined AXI
- Less of an interconnect/topology more of a protocol
- AXI – eXtensible Interconnect
  - Backward compatible with AHB/APB.
  - ARM11 (+)
  - 77 control signals (27 for AHB and 4 for APB)
  - 1-16 masters, 1-16 slaves.
  - Interconnect with crossbar, 5 channels.
  - Supports simultaneous read/write



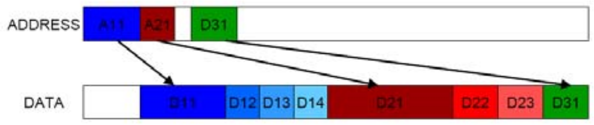
AXI Burst Write

Zürcher Fachhochschule 41

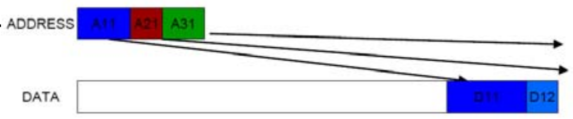
AXI Burst Write: The clou here is that there is a single address given to the slave and that there is auto incrementing of addresses as the data is being written into the slave. Contemplating this allows us to think in terms of channels and indeed above we see five channels, Write Address/Control (ARW, Write Data Channel (W), Write response channel (B) and by proxy Read Address Channel (AR) and Read Data Channel (R), the letters in brackets are the signal prefixes. By dividing up the bus into five channels and assuming that these are asynchronous to each other some interesting properties may be observed.

The first is that simultaneous read and write is possible. This is made possible by defining a read address channel, the read and write data channels were already split in the AHB specification.

The second is that by assuming a single address for a burst address, addresses can be properly queued (see figure below). This means that the timing of the address bus may be held independentl



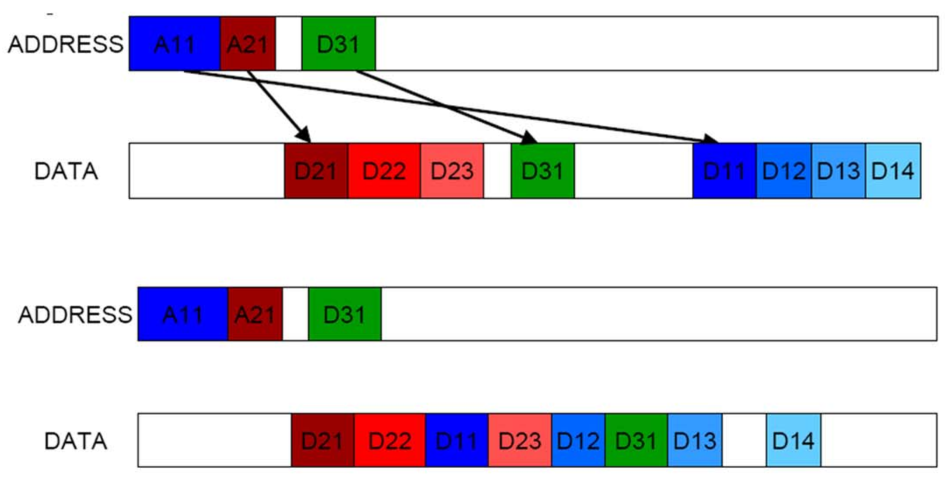
There is a slight problem



erything is held up

AXI Out-of-Order Completion/Interleaving

Zurich University of Applied Sciences
zhaw School of Engineering
InES Institute of Embedded Systems

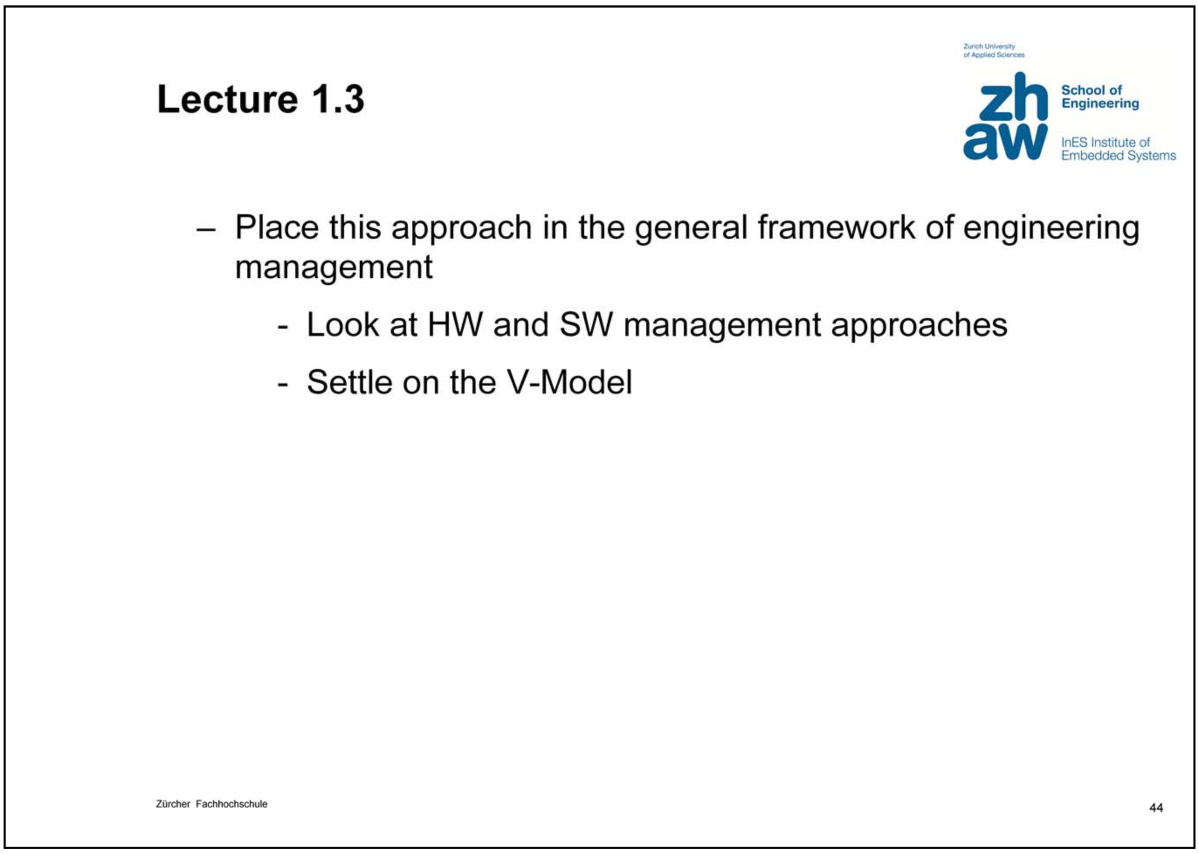


Zürcher Fachhochschule 42

Out-of-Order Completion: By imparting the ability to queue address, essentially transaction requests – interleaving is possible allowing maximum usage of data channel bandwidth. In fact measurements show that AHB offers less than 50% utility whereas AXI up-to 80% efficiency.

Conclusions

- AMBA is a set of growing specifications.
- AHB is the “standard” high performance bus, APB that for slow peripherals.
- AHP enhanced by multi-layer AHB – new topology concepts applied to try and minimise negative effects of shared-bus topologies
- AXI developed to asynchronise the bus, that is split it up into receive and transmit components, reduce timing effects and increase bus utilisation, at the expense of more expensive (?) slaves and switching technology.



Lecture 1.3

Zurich University of Applied Sciences

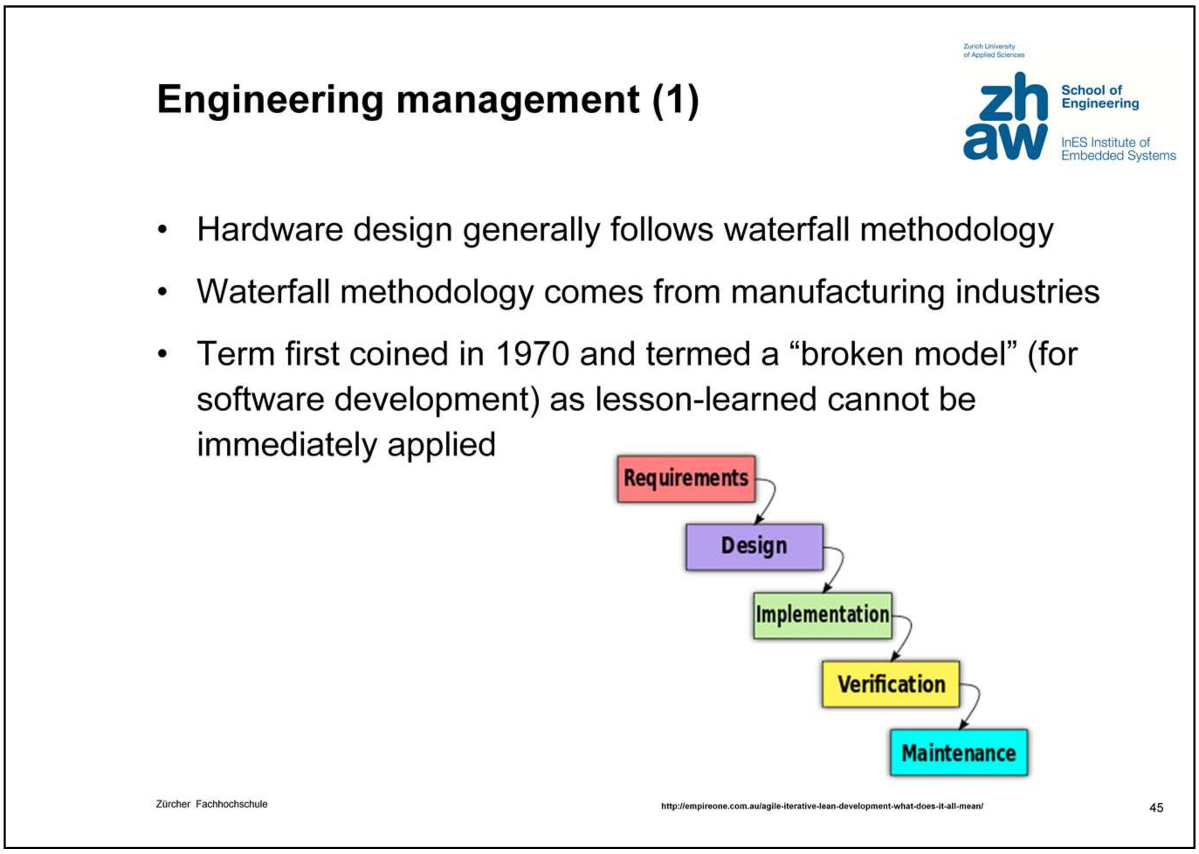
School of Engineering

InES Institute of Embedded Systems

- Place this approach in the general framework of engineering management
  - Look at HW and SW management approaches
  - Settle on the V-Model

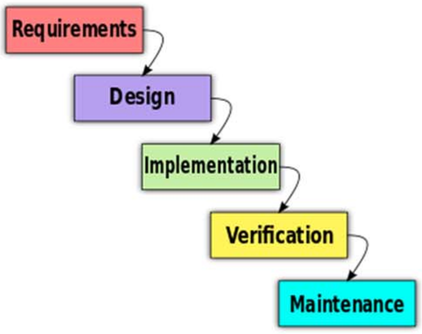
Zürcher Fachhochschule

44



Engineering management (1)

- Hardware design generally follows waterfall methodology
- Waterfall methodology comes from manufacturing industries
- Term first coined in 1970 and termed a “broken model” (for software development) as lesson-learned cannot be immediately applied



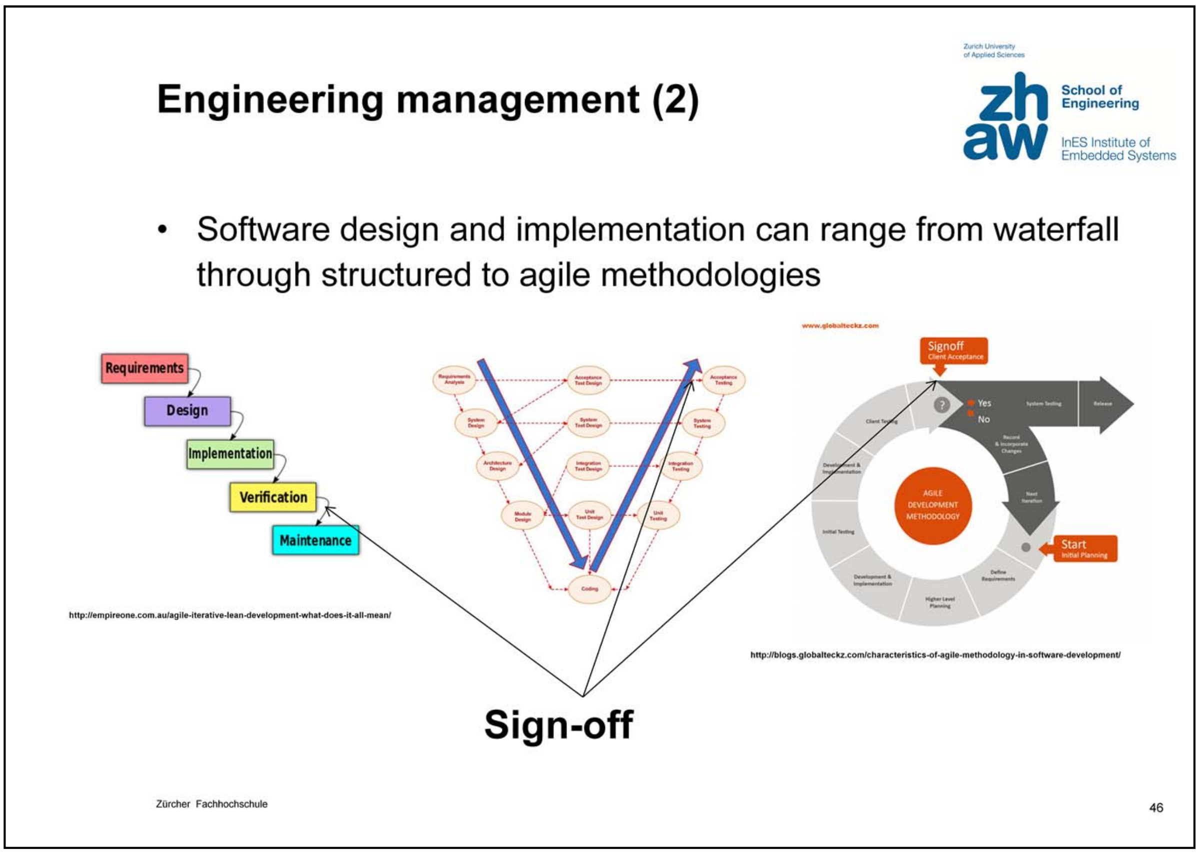
Zürcher Fachhochschule http://empireone.com.au/agile-iterative-lean-development-what-does-it-all-mean/ 45

The concept of the waterfall methodology originates from the manufacturing industry but was first coined as a phrase in 1970 in a paper by Dr. Winston W. Royce in an influential paper on the additional complexity faced by software engineers and that a new development methodology was required.

http://www.cs.umd.edu/class/spring2003/cmsc838p/Process/waterfall.pdf

The issues lie in the fact that in software engineering the primary cost is developer man-hours. In other industries (ship-building for instance) the primary cost is manufacturing hours and materials. Therefore it is expected that a static development/implementation plan can be followed from the day the specifications are set down (“Keel has been laid, height of toilet roll holder cannot be changed”). Hardware development is similar in that much of developer hours is spent in finding cheaper ways to produce the electronics and reduce purchase costs of components. This has the effect of disciplining the engineer and subjecting him to (well accepted) formal processes.

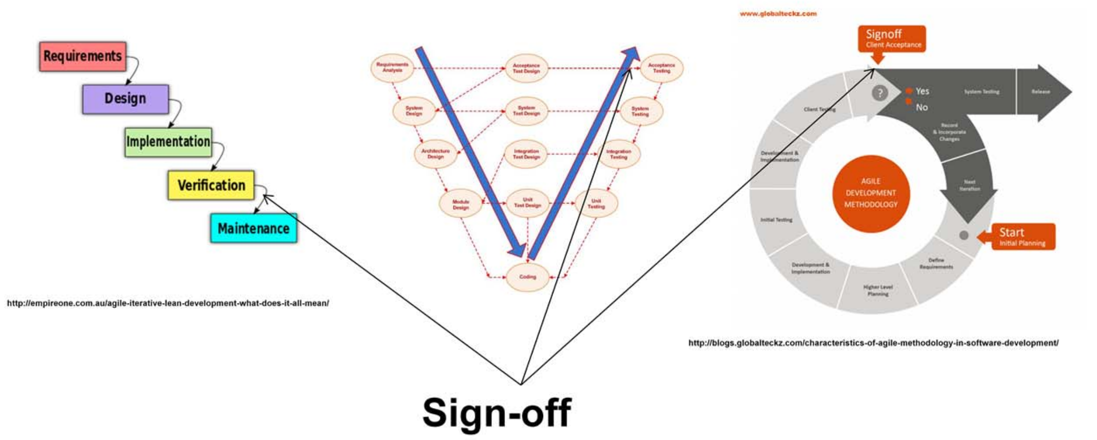
Software doesn’t suffer these constraints – post release fixes are (erroneously) considered cheap. The discipline (if any) imposed on programmers is therefore different.



Engineering management (2)

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems

- Software design and implementation can range from waterfall through structured to agile methodologies



http://empireone.com.au/agile-iterative-lean-development-what-does-it-all-mean/

http://blogs.globalteckz.com/characteristics-of-agile-methodology-in-software-development/

Zürcher Fachhochschule 46

A second issue with the nature of the software business is that much of it is bespoke – this means that the client is more prominent than in other disciplines. In particular there is an assumption that because the client is generally not technically orientated – "he doesn't know what he wants" and therefore the specifications are liable to change during the lifetime of the development project.

An internal problem with software development is that because these stretch over a period of time and because software is more fluid than the results of other engineering activities. Not having understood the power demands of an op-amp properly may result in a hardware re-design costing 30k or so but the actual change is generally minor and involves one maybe two engineers. A flaw in the interface of a software package can mean an entire team spends a month or two re-writing code and much that was written before is lost.

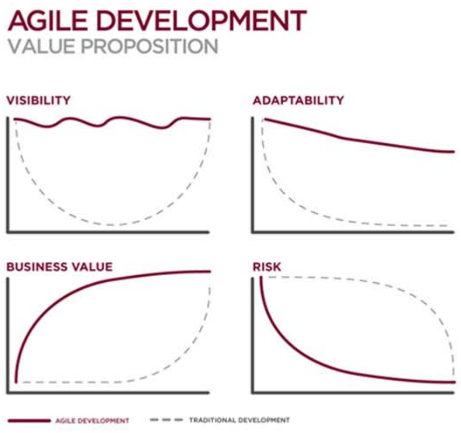
Therefore there is a case to be made for "embracing" these issues and developing a methodology that allows "on the job learning" to be integrated in the end product. This can, and frequently does, include acceptance of writing code that will be thrown away. These methodologies are known as agile methodologies.

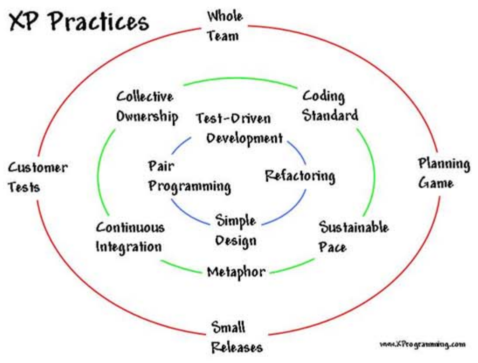
Credit middle picture from: Lost in the mists of time – Wikipedia I think but haven't been able to find it again.

Agile methodology

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems

- Agile methods create value by extreme iteration



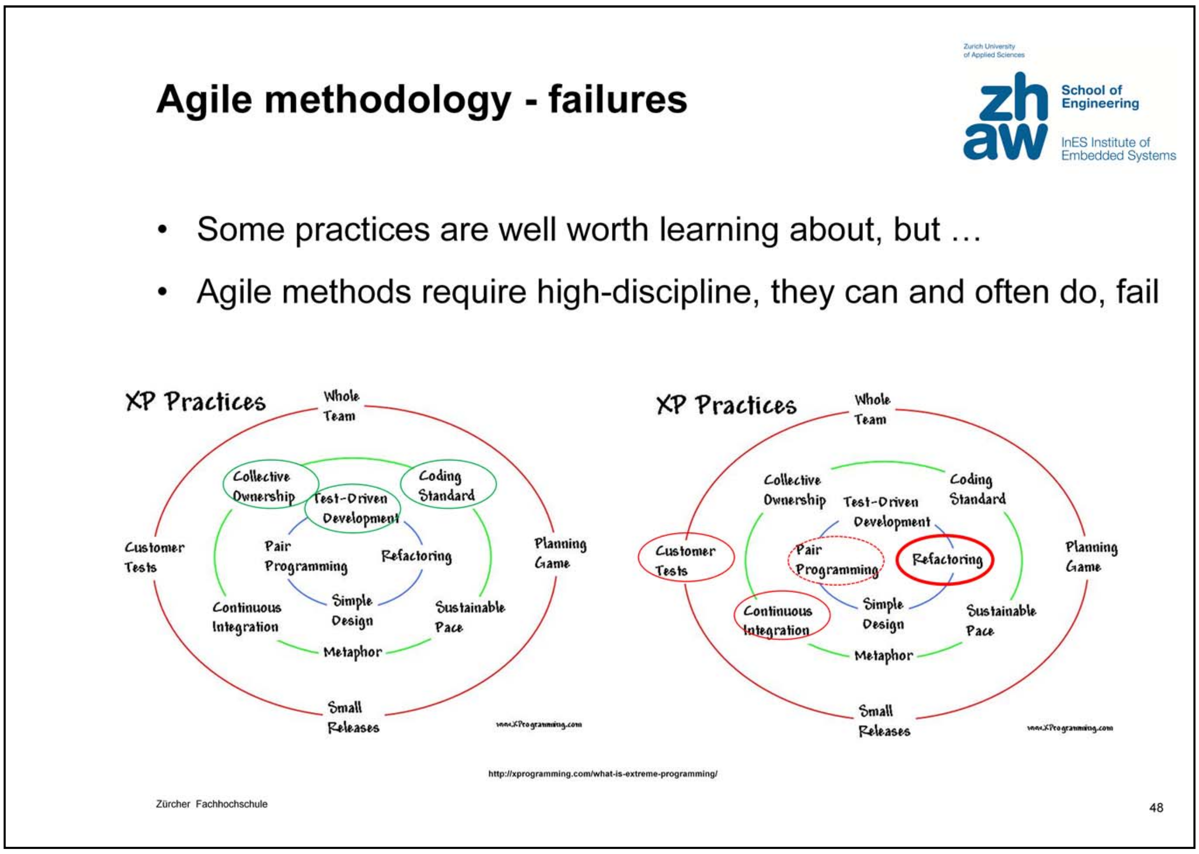


http://xprogramming.com/what-is-extreme-programming/

Picture credit: http://empireone.com.au/agile-iterative-lean-development-what-does-it-all-mean/

Zürcher Fachhochschule 47

Business proposition of large numbers of iterations. eXtreme programming, one of the first proponents of agile methodologies managed to discredit itself by overuse of the re-factoring practice, but it was made up of several best-practices which in themselves are worth closer attention.



Agile methodology - failures

- Some practices are well worth learning about, but …
- Agile methods require high-discipline, they can and often do, fail

http://xprogramming.com/what-is-extreme-programming/

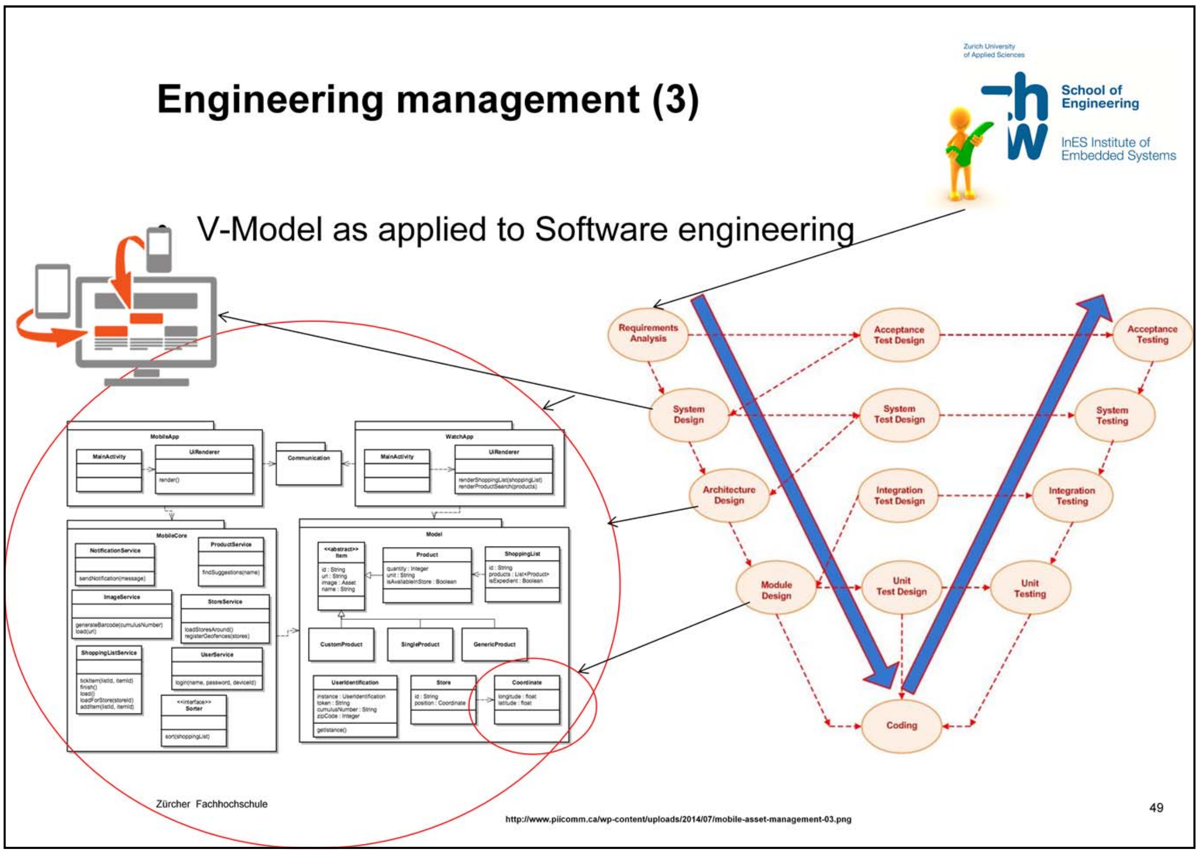
Zürcher Fachhochschule 48

The positives of XP – in the lecturers experience – are the use of coding standards (standardisation of what and how the programmers produce code) collective ownership (every body can change it – only works if coding standards are used) and test-driven development. In the lecturers opinion – and he once (expensively) applied iterative practices to HW development – the lesson HW developers could learn from XP is test-driven development, i.e. don't design anything until you know how you are going to test it.

The failures of XP are well known: XP demands a customer on-site to write use stories and to acceptance test finished features, its expensive but what it does is to place the entire responsibility for the project onto the customer. Engineers are supposed to have a sense of professionalism which includes taking responsibility for their actions. XP eventually failed as one of the main coders wouldn't stop re-factoring. Pair programming is a thing (supposed to increase code quality) which not all engineers can adapt to and was one of the most frequently cited issues with XP. In essence, design-reviews are the penultimate pair programming activity.

In terms of hardware development one of the killer practices is continuous integration. Either the hardware works or it doesn't. Since the test-driven development practice means that tests need to be written first the methodology fails when dealing with I/O, this includes GUI's and data-bases. It fails mainly because the tests become so expensive to create and because often in the average embedded engineering project the HW isn't available. It puts a dampener on the iterative process.

Because of these, and other reasons, agile processes are less suited to the management of embedded projects. If a process is used at all, which in many cases it is not, then the V-Model has become the most favoured.



Engineering management (3)

Zurich University of Applied Sciences

School of Engineering

InES Institute of Embedded Systems

- V-Model as applied to Software engineering

Zürcher Fachhochschule

http://www.piicomm.ca/wp-content/uploads/2014/07/mobile-asset-management-03.png

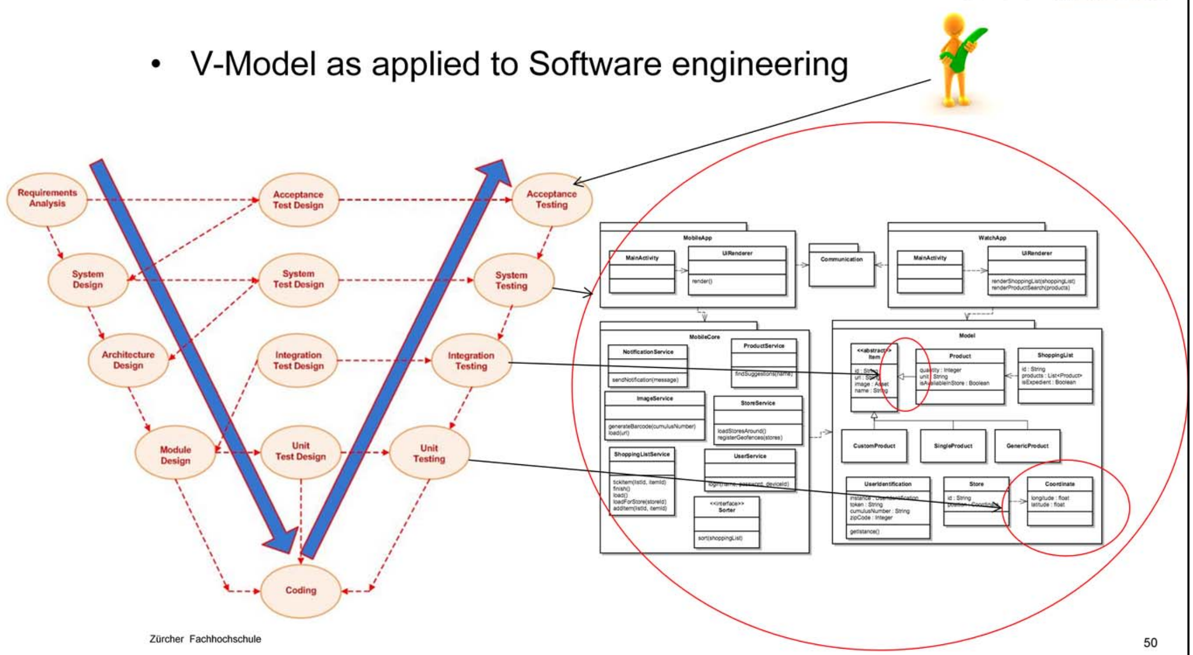
49

http://support.sas.com/resources/papers/proceedings11/124-2011.pdf is a good basic introduction to the V-Model.

Engineering management (4)

Zurich University of Applied Sciences
zh aw School of Engineering
InES Institute of Embedded Systems

- V-Model as applied to Software engineering



Zürcher Fachhochschule

50

http://partnerswithyou.co.uk/wp-content/uploads/2012/12/shutterstock_customer-service.jpg

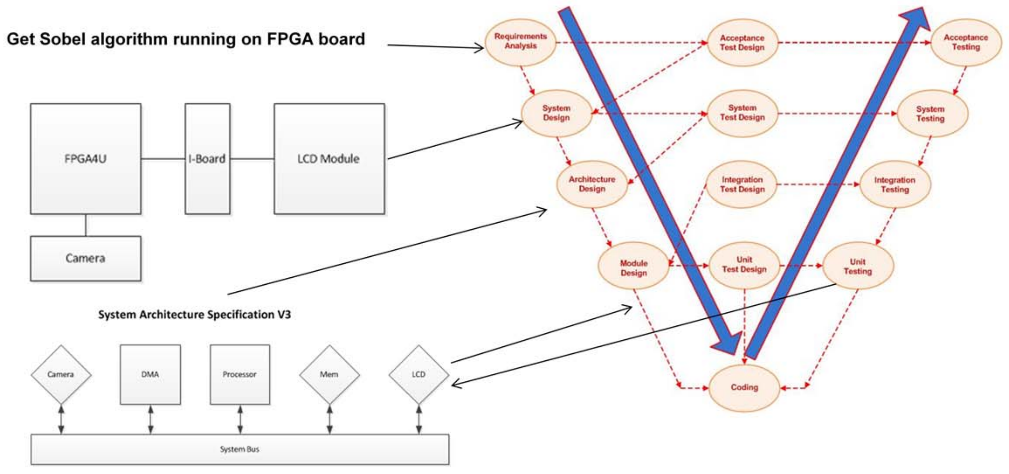
http://support.sas.com/resources/papers/proceedings11/124-2011.pdf is a good basic introduction to the V-Model.

Zurich University of Applied Sciences

zh aw School of Engineering InES Institute of Embedded Systems

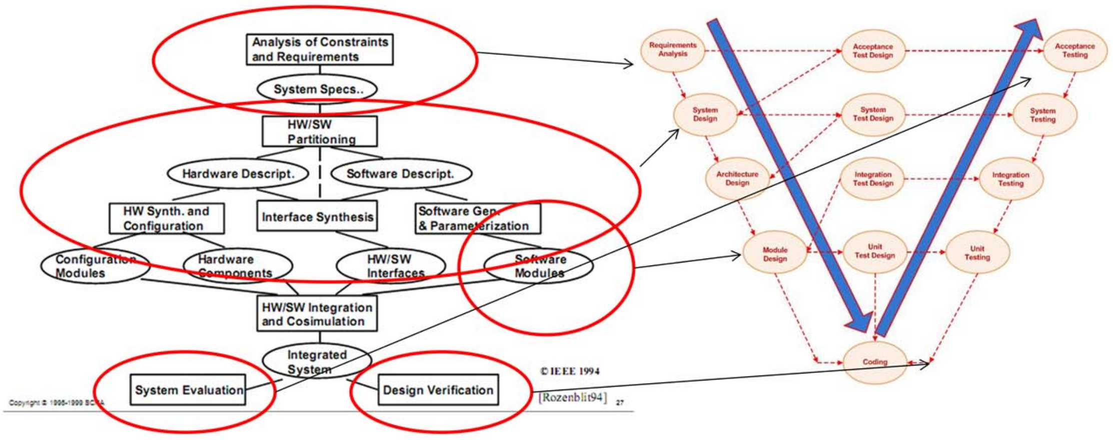
Engineering management (5)

- V-Model as applied to HW/SW deliverables



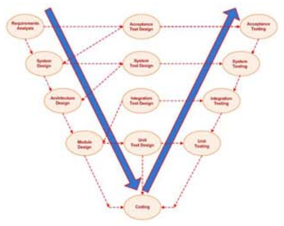
Engineering management (6)

- V-Model as applied to HW/SW co-design



Engineering management (7)

- Conclusions
  - V-Model is a framework, it can be used for HW, SW and any combinations thereof
  - V-Model can be part of a process – this is what ISO 900x certifies
  - V-Model is an essential process in safety related development
  - "V-Model is a useful analytical tool for determining the cost/state of a project



Conclusions

- Structured approach to design decisions -> scientific approach,
  - Thesis -> experiment -> verifiy
- HW-SW co-design techniques provide a structured approach and language
- These techniques must be embedded in some management structure/process
- If industry has an opinion then it tends towards V-model