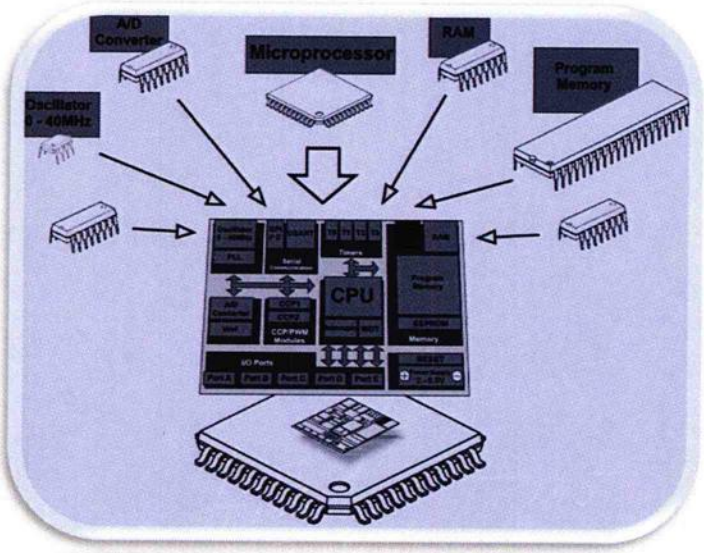
Mittuniversitetet
MID SWEDEN UNIVERSITY

Försättsblad Prov Original

Kurskod	Provkod	Tentamensdatum
ET011G	T200	2019-01-15
Kursnamn	Elektroteknik GR (A), Mikrodatorteknik	
Provnamn	Tentamen	
Ort	Sundsvall	
Termin		
Ämne		

6895642268

Utskriven 2019-01-14 kl. 14:35:23



Exam in Microprocessor Technology

2019-01-15

Writing time:	5 hours
Max points:	49
Minimum points for pass:	24
Responsible teacher:	Johan Sidén (070 - 671 71 71)
Aids:	Calculator

OBS!
Don't forget to show all steps in your clear calculations, constructions and own figures! Motivate all conclusions! Prove that you have understood! If you find that information in assignments is not complete you will make a well thought through assumption! Answers are written in English or Swedish.

Good Luck!

1. The von Neumann computer

a) The von Neumann computer executes in five stages. What does this mean?

b) Describe with a simple sketch / drawing how a pipeline works in this five stage process.

Stage Number

	1	2	3	4	5	6	7	8	9	10
IF	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
ID		I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9
OF			I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8
EX				I_1	I_2	I_3	I_4	I_5	I_6	I_7
RS					I_1	I_2	I_3	I_4	I_5	I_6

IF = Instruction Fetch
ID = Instruction Decode
OF = Operand Fetch
EX = Exekvera
RS = Result Store

I_x=Instruction number x

5+3=8p

2. It's common to talk about two fundamental computer architectures, von-Neumann and Harvard. Describe the difference between these two architectures

8p

3. The CPU core of the AVR32 has five important register types. List and describe the function of these registers.

5p

4. Convert 0.7 to a binary number. Show the arithmetic steps you have used.

3p

5. Describe with a simple example the meaning of an interrupt and give advantages and disadvantages with interrupts as compared to polling.

8p

6. In an asynchronous communication system a hand shaking algorithm is normally used to synchronize a data transfer. Describe with a simple example how a data transfer from point A to B is synchronized with aid of a hand shaking algorithm. What controls signals are necessary?

6p

7. RAM

a) Discuss the differences between SRAM and DRAM, including their suitable application areas.

b) What differs as SDRAM from a "normal DRAM"?

6+2=8p

8. How many signal lines are needed for a bus of the type

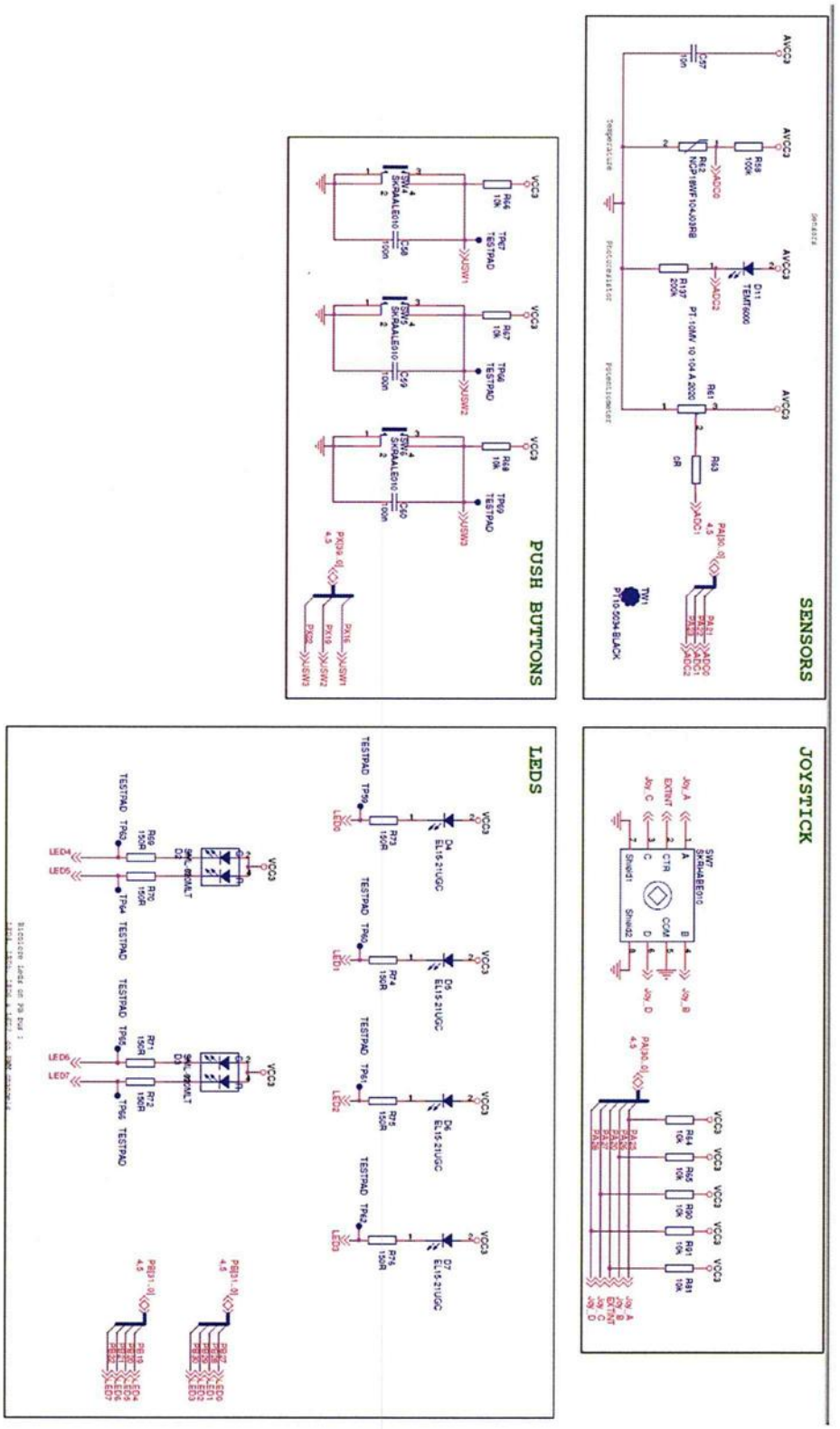
a) CAN

b) USB

c) I2C

3p

Appendix I – EVK1100 Schematics



Appendix II – AVR32 Preliminary

12. Peripherals

12.1 Peripheral address map

Table 12-1. Peripheral Address Mapping

Address		Peripheral Name	Bus
0xE0000000	USBB	USBB Slave Interface - USBB	HSB
0xFFFE0000	USBB	USBB Configuration Interface - USBB	PBB
0xFFFE1000	HMATRIX	HMATRIX Configuration Interface - HMATRIX	PBB
0xFFFE1400	FLASHC	Flash Controller - FLASHC	PBB
0xFFFE1800	MACB	MACB Configuration Interface - MACB	PBB
0xFFFE1C00	SMC	Static Memory Controller Configuration Interface - SMC	PBB
0xFFFE2000	SDRAMC	SDRAM Controller Configuration Interface - SDRAMC	PBB
0xFFFF0000	PDCA	Peripheral DMA Interface - PDCA	PBA
0xFFFF0800	INTC	Interrupt Controller Interface - INTC	PBA
0xFFFF0C00	PM	Power Manager - PM	PBA
0xFFFF0D00	RTC	Real Time Clock - RTC	PBA
0xFFFF0D30	WDT	WatchDog Timer - WDT	PBA
0xFFFF0D80	EIC	External Interrupt Controller - EIC	PBA
0xFFFF1000	GPIO	General Purpose IO Controller - GPIO	PBA
0xFFFF1400	USART0	Universal Synchronous Asynchronous Receiver Transmitter - USART0	PBA
0xFFFF1800	USART1	Universal Synchronous Asynchronous Receiver Transmitter - USART1	PBA

Table 12-9. GPIO Controller Function Multiplexing

TQFP100	VQFP144	PIN	GPIO Pin	Function A	Function B	Function C
7	11	PB24	GPIO 56	TC - B0	USART1 - DSR	
8	13	PB25	GPIO 57	TC - A1	USART1 - DTR	
9	14	PB26	GPIO 58	TC - B1	USART1 - RI	
10	16	PB27	GPIO 59	TC - A2	PWM - PWM[4]	
14	19	PB28	GPIO 60	TC - B2	PWM - PWM[5]	
15	20	PB29	GPIO 61	USART2 - RXD	PM - GCLK[1]	EBI - NCS[2]
16	21	PB30	GPIO 62	USART2 - TXD	PM - GCLK[2]	EBI - SDCS
17	22	PB31	GPIO 63	USART2 - CLK	PM - GCLK[3]	EBI - NWAIT
83	85	PC00	GPIO 64			
84	86	PC01	GPIO 65			
85	124	PC02	GPIO 66			
86	125	PC03	GPIO 67			
93	132	PC04	GPIO 68			
94	133	PC05	GPIO 69			
	1	PX00	GPIO 100	EBI - DATA[10]	USART0 - RXD	
	2	PX01	GPIO 99	EBI - DATA[9]	USART0 - TXD	
	4	PX02	GPIO 98	EBI - DATA[8]	USART0 - CTS	
	10	PX03	GPIO 97	EBI - DATA[7]	USART0 - RTS	
	12	PX04	GPIO 96	EBI - DATA[6]	USART1 - RXD	
	24	PX05	GPIO 95	EBI - DATA[5]	USART1 - TXD	
	26	PX06	GPIO 94	EBI - DATA[4]	USART1 - CTS	
	31	PX07	GPIO 93	EBI - DATA[3]	USART1 - RTS	
	33	PX08	GPIO 92	EBI - DATA[2]	USART3 - RXD	
	35	PX09	GPIO 91	EBI - DATA[1]	USART3 - TXD	
	38	PX10	GPIO 90	EBI - DATA[0]	USART2 - RXD	
	40	PX11	GPIO 109	EBI - NWE1	USART2 - TXD	
	42	PX12	GPIO 108	EBI - NWE0	USART2 - CTS	
	44	PX13	GPIO 107	EBI - NRD	USART2 - RTS	
	46	PX14	GPIO 106	EBI - NCS[1]		TC - A0
	59	PX15	GPIO 89	EBI - ADDR[19]	USART3 - RTS	TC - B0
	61	PX16	GPIO 88	EBI - ADDR[18]	USART3 - CTS	TC - A1
	63	PX17	GPIO 87	EBI - ADDR[17]		TC - B1
	65	PX18	GPIO 86	EBI - ADDR[16]		TC - A2
	67	PX19	GPIO 85	EBI - ADDR[15]	EIM - SCAN[0]	TC - B2
	87	PX20	GPIO 84	EBI - ADDR[14]	EIM - SCAN[1]	TC - CLK0
	89	PX21	GPIO 83	EBI - ADDR[13]	EIM - SCAN[2]	TC - CLK1
	91	PX22	GPIO 82	EBI - ADDR[12]	EIM - SCAN[3]	TC - CLK2
	95	PX23	GPIO 81	EBI - ADDR[11]	EIM - SCAN[4]	
	97	PX24	GPIO 80	EBI - ADDR[10]	EIM - SCAN[5]	

22.5 General Purpose Input/Output (GPIO) User Interface

The GPIO controls all the I/O pins on the AVR32 microcontroller. The pins are managed as 32-bit ports that are configurable through an PB interface. Each port has a set of configuration registers. The overall memory map of the GPIO is shown below. The number of pins and hence the number of ports is product specific.

In the Peripheral muxing table in the Peripherals chapter each GPIO line has a unique number. Note that the PA, PB, PC and PX ports do not directly correspond to the GPIO ports. To find the corresponding port and pin the following formulas can be used:

GPIO port = floor((GPIO number) / 32), example: floor((36)/32) = 1

GPIO pin = GPIO number mod 32, example: 36 mod 32 = 4

The table below shows the configuration registers for one port. Addresses shown are relative to the port address offset. The specific address of a configuration register is found by adding the register offset and the port offset to the GPIO start address. One bit in each of the configuration registers corresponds to an I/O pin.

Table 22-2. GPIO Register Map

Offset	Register	Function	Name	Access	Reset value
0x00	GPIO Enable Register	Read/Write	GPER	Read/Write	1b for each implemented GPIO pin in port
0x04	GPIO Enable Register	Set	GPERS	Write-Only	
0x08	GPIO Enable Register	Clear	GPERC	Write-Only	
0x0C	GPIO Enable Register	Toggle	GPERT	Write-Only	
0x10	Peripheral Mux Register 0	Read/Write	PMR0	Read/Write	0x00000000

Table 22-2. GPIO Register Map

Offset	Register	Function	Name	Access	Reset value
0x14	Peripheral Mux Register 0	Set	PMR0S	Write-Only	
0x18	Peripheral Mux Register 0	Clear	PMR0C	Write-Only	
0x1C	Peripheral Mux Register 0	Toggle	PMR0T	Write-Only	
0x20	Peripheral Mux Register 1	Read/Write	PMR1	Read/Write	0x00000000
0x24	Peripheral Mux Register 1	Set	PMR1S	Write-Only	
0x28	Peripheral Mux Register 1	Clear	PMR1C	Write-Only	
0x2C	Peripheral Mux Register 1	Toggle	PMR1T	Write-Only	
0x30	RESERVED	-	-	-	
0x34	RESERVED	-	-	-	
0x38	RESERVED	-	-	-	
0x3C	RESERVED	-	-	-	
0x40	Output Driver Enable Register	Read/Write	ODER	Read/Write	0x00000000
0x44	Output Driver Enable Register	Set	ODERS	Write-Only	
0x48	Output Driver Enable Register	Clear	ODERC	Write-Only	
0x4C	Output Driver Enable Register	Toggle	ODERT	Write-Only	
0x50	Output Value Register	Read/Write	OVR	Read/Write	0x00000000
0x54	Output Value Register	Set	OVRS	Write-Only	
0x58	Output Value Register	Clear	OVRC	Write-Only	
0x5c	Output Value Register	Toggle	OVRT	Write-Only	
0x60	Pin Value Register	Read	PVR	Read-Only	depending on pin states
0x64	Pin Value Register	-	-	-	
0x68	Pin Value Register	-	-	-	
0x6c	Pin Value Register	-	-	-	
0x70	Pull-up Enable Register	Read/Write	PUER	Read/Write	0x00000000
0x74	Pull-up Enable Register	Set	PUERS	Write-Only	
0x78	Pull-up Enable Register	Clear	PUERC	Write-Only	
0x7C	Pull-up Enable Register	Toggle	PUERT	Write-Only	
0x80	Open Drain Mode Enable Register	Read/Write	ODMER	Read/Write	0x00000000
0x84	Open Drain Mode Enable Register	Set	ODMERS	Write-Only	
0x88	Open Drain Mode Enable Register	Clear	ODMERC	Write-Only	
0x8C	Open Drain Mode Enable Register	Toggle	ODMERT	Write-Only	
0x90	Interrupt Enable Register	Read/Write	IER	Read/Write	0x00000000
0x94	Interrupt Enable Register	Set	IERS	Write-Only	
0x98	Interrupt Enable Register	Clear	IERC	Write-Only	
0x9C	Interrupt Enable Register	Toggle	IERT	Write-Only	