The conditions is a second of the impact substitute of them & then pure there "value" on to the Q LDDs on DEL-SeC — poor \$ a condition to the LDD on "memory mapped" to address on the second of the LDD on The Condition of the LDD on The Condition of the Condition of the LDD on The Condition of t		
Description of the speciment of the impute surfeches of them & then puts that "value" on to the 9 LEDs on DEA. Suc. — part 5 String that the LEDs are "memory-mapped" to address on the impute surfeches of them & then puts that "value" on to the 9 LEDs on DEA. Suc. — part 5 String that the LEDs are "memory-mapped" to address on the impute surfeches of "machine code" numbers. Steinbig-language Program use "address on the impute surfeches of its into a degreeous of "machine code" numbers. Jedifore LED. ADDRESS On 1600 Jed modeline LED. ADDRESS On 1600 Jed modeline LED. ADDRESS On 1600 Jed modeline String on 1600 J		
The support of the state of the support of the supp		
is a address. Compared by the school Comp	DOUT -	
The Confidence of the Confiden	w_∸	
De Cupper à bits social De Cupper à bits soci		
and a program to touch the values on the injures swintches 9 of them & they puts that "value" on to the 9 LEDs on DE1-ScC	s @ address	
ant a program so trace the values on the inputs switches 9 of them & then puts that "value" on to the 9 LEDs on DE1-SaC - port & street that the LEDs are "memory -mapped" to Oxfores 0 or 10000 Stembly-language Program: use "assembler" to varieste it into A sequence of "machine code members. define LED_ADDRESS Ox1000	o. upper 4 b	s = 0001
assume that the LED; are "memory -mapped" to address 0 x 1000 Ca 3000 Ca		
assume that the LEDs are "memory -mapped" to address 0 x 1000 Ca 3000 Ca		
Sumbley-language Program use assembler' to trouslate it into A sequence of machine code manbers. -define LED_RDDRESS 0,1000 but executed. -define SN-ADDRESS 0,1000 but executed. -define SN-ADDRESS 0,1000 but executed. -define SN-ADDRESS 0,1000 but executed.	ant a program	take the values on the inputs switches-9 of them & then puts that "value" on to the 9 LEDs on DE1-SoC part 3
Sumbley-language Program use assembler' to trouslate it into A sequence of machine code manbers. -define LED_RDDRESS 0,1000 but executed. -define SN-ADDRESS 0,1000 but executed. -define SN-ADDRESS 0,1000 but executed. -define SN-ADDRESS 0,1000 but executed.	ssume that th	EDs are "memory-mapped" to address 0x1000
Seemble - language Programs use assembler to maristate it into a sequence of machine code mumbers. . define LED_ADDRESS On 1000 not executed define SW_ADDRESS On 1000 just makes program easier to sorter & read for humans myt 70, \$LLD_ADDRESS TA+On 1000 just makes program easier to sorter & read for humans myt 70, \$LLD_ADDRESS TA+On 1000 just makes program easier to sorter & read for humans myt 70, \$LLD_ADDRESS TA+On 1000 just makes program easier to sorter & read for humans myt 70, \$LLD_ADDRESS TA+On 1000 just makes in ras 0,23000 st 70, \$LTA] TA+On 1000 just makes in ras 0,23000 st 70, \$LTA] TA+On 1000 just makes in ras 0,23000 my 71, \$LLD_ADDRESS TA+On 1000 just makes in ras 0,23000 my 71,		
define SU_ADDRESS 0x1000 not executed. define SU_ADDRESS 0x1000 just makes program easier to tarke & read for humans myt ro. \$1LED.ADDRESS		
define SW_ADDRESS 0x 3000 just makes program easier to write & read for humans my 12. \$1.ED.ADDRESS 1x. \$2.0x300 just makes program easier to write & read for humans my 12. \$1.ED.ADDRESS 1x. \$2.0x300 just makes in 1x. \$2.0x300 just mak	ssembly-langua	e Program: use "assembler" to translate it into a sequence of "machine code" numbers.
define SW_ADDRESS 0x 3000 just makes program easier to write & read for humans my 12. \$1.ED.ADDRESS 1x. \$2.0x300 just makes program easier to write & read for humans my 12. \$1.ED.ADDRESS 1x. \$2.0x300 just makes in 1x. \$2.0x300 just mak	define L	ADDRESS Ox1000 not executed.
my ty, 8 LED. ADDRESS Ta + Date to 1		
my r. # SLE_ADDRESS r. * On address in re_0.2000 11. Icl re, [ra] re data from address in re_0.2000 12. re, [ra] re data from address in re_0.2000 13. re, [ra] address in ra (LEDs) * re 14. re, [ra] address in ra (LEDs) * re 15. re, [ra] address in ra (LEDs) * re 16. re, [ra] address in ra (LEDs) * re 16. re, [ra] address in ra (LEDs) * re 17. * 20 or my PC, # Alin or b # Alin creates on infinite loop of 2 load & store instruction to be executed will be fetched from address a 16. representational branches * do the branch only if some condition is true 16. conditional branches * do the branch only if some condition is true 16. conditional branches * do the branch only if some condition is true 16. adapte: the value in a register is zero 16. adapte: the value in a register is zero 16. adapte: the value in a register is zero 16. adapte: the value in a register is zero 16. adapte: the value in a register is zero 16. adapte: the value in a register is zero 17. adapte: the value in a register is zero 18. adapte: the value in a	-define Su	ADDRESS Ox 5000 just makes program easier to write it read for numans
my r. # SLE_ADDRESS r. * On address in record and recor		
my: To, 485W_ADDRESS To Documents of the program address in To (LEDS) 4 To is led To, (Tra) To 4 class from address in To (LEDS) 4 To my: Tr, \$2 or my: PC, \$2 DC \$T 4 2. Sets the program counter to 2 anext instruction to be executed will be fetched from address at or my: PC, 4HAIN or b. \$MAIN creates an infinite loop of 2 load & store instruction bere are both unconditional bronches (e.g. b. \$MAIN) conditional bronches -> do the bronch only if some condition is true. complete the value in G. register is zero beq \$MAIN go to \$MAIN if 2s my: Tr, Tr, other use, just go to next instruction 2		SIED DEDDETS
is Id Fo. [Tra] To 4 data from address in To 1220000 st To. [Tra] address in To 1220 + To mV Tr. \$2 or mv PC, \$2 PC = Tr 4 2. Sets the program counter to 2 enext instruction to be executed will be fetched from address 2 or mv PC. \$HAIN or b \$HAIN		
The proposed of the proposed country of a personal process of the program country of a personal process of the process of the program country of a personal process of the process of the program country of a personal process of the process of the process of the program country of a personal process of the	mvt Y	#SW_ADDRESS TA * On A OD O
my Pr. #2 or my Pc. #2 Or my Pc. #MAIN or b #MAIN Creates an infinite loop of 2 load & store instruction to be executed will be fetched from address a or my Pc. #MAIN or b #MAIN creates an infinite loop of 2 load & store instruction bere are both unconditional branches —edo the branch only if some condition is true. cample: the value in 0. register is zero beq #MAIN go to #MAIN if 25 brown and main if 25 bcc #MAIN go to #MAIN if 25 bcc #MAIN go to #MAIN if 25 bcs #MAIN go to #MAIN if 25 bcs #MAIN go to #MAIN if 25 condition is different: bicond? #D condition: ooo hone cond cond iii 1 xxxx DDDDDDDDDD condition color oo o	: ld r	[re] re+data from address in r4=0x3000
my Pr. #2 or my Pc. #2 Or my Pc. #MAIN or b #MAIN Creates an infinite loop of 2 load & store instruction to be executed will be fetched from address a or my Pc. #MAIN or b #MAIN creates an infinite loop of 2 load & store instruction bere are both unconditional branches —edo the branch only if some condition is true. cample: the value in 0. register is zero beq #MAIN go to #MAIN if 25 brown and main if 25 bcc #MAIN go to #MAIN if 25 bcc #MAIN go to #MAIN if 25 bcs #MAIN go to #MAIN if 25 bcs #MAIN go to #MAIN if 25 condition is different: bicond? #D condition: ooo hone cond cond iii 1 xxxx DDDDDDDDDD condition color oo o		Eral addition in the Land Ara
Der NY PC, SHAIN or b SMAIN Creates on infinite loop of 2 load & store instruction iconditional branches — do the branch only if some condition is true conditional branches — do the branch only if some condition is true comple: the value in G register is zero beq SMAIN go to SMAIN if Zzi WY T., To other wise, just go to new instruction bec SMAIN go to SMAIN if Zzo cond ini i xxxx DDDDDDDDD cond cond ini i xxxx DDDDDDDDDD cond		
Deep and both unconditional branches (e.g. b &HARIN) conditional branches — do the branch only if some condition is true ample: the value in G register is zero beq \$MARIN go to \$MARIN if Zz mv Y., Yz other usse, just go to next instruction bec. \$MARIN go to \$MARIN if Zz cond it i xxx DDDDDDDDD cond ooi eq ooi eq ooi eq ooi oc and it i xxx DDDDDDDDD cond cond it i xxx DDDDDDDDD cond ooi co	mV T2, #:	or my PC. 22 PC=Y1 < 2. Sets the program counter to a * next instruction to be executed will be fetched from address a
Deep and both unconditional branches (e.g. b &HARIN) conditional branches — do the branch only if some condition is true ample: the value in G register is zero beq \$MARIN go to \$MARIN if Zz mv Y., Yz other usse, just go to next instruction bec. \$MARIN go to \$MARIN if Zz cond it i xxx DDDDDDDDD cond ooi eq ooi eq ooi eq ooi oc and it i xxx DDDDDDDDD cond cond it i xxx DDDDDDDDD cond ooi co	or my Pc, #M	Nor b #MAIN creates on infinite loop of 2 load & store instruction
Econditional branches > do the branch only if some condition is true Complet the value in G register is zero beq SMAIN go to SMAIN if Z=1 MV Y., Ya other wise, just go to new instruction 2 bne SMAIN go to SMAIN if Z=0 bcc SMAIN go to SMAIN if C=1 bcc SMAIN go to SMAIN if C=1 cooling is different: bicondi sD condition: 000 none cond HI I XXX DDDDDDDDD OII CC OII		
Londitional branches - do the branch only if some condition is true Lample: the value in G register is zero beq shall go to shall if z=1 mv Y., Y. and the mise, just go to new instruction bec shall go to shall if z=0 bec shall go to shall if z=0 bes shall go to shall if z=0 bes shall go to shall if z=1 secoling is different: bicond; ab condition: 000 none cond iii 1 xxx ddifferent: bicond; ab condition: 000 none oii cc shall in xxx ddifferent: bicond; ab condition: 000 none oii cc oii cc oii cc oii cc oii cc oii cc shall in xxx ddifferent: bicond; ab condition: 000 none oii cc oii cc oii cc oii cc oii cc oii cc shall in xxx ddifferent: bicond; ab condition: 000 none oii cc		
beq #MRIN go to #MRIN if 2.2 beq #MRIN go to #MRIN if 2.2 bec #MRIN go to #MRIN if 2.2 bcc #M	vere are both	conditional branches (e.g. b eMAIN)
beq #MAIN go to #MAIN if Zz beq #MAIN go to #MAIN if Zz	conditional	whether - bille the branch could be some condition is true
beq smain go to smain if zz B		
B	cample: the va	e in G. register is zero
B	-57 br	#MAIN 00 10 8MAIN # 2-1
B		
bec \$MAIN go to \$MAIN if cal bes \$MAIN go to \$MAIN if cal bes \$MAIN go to \$MAIN if cal becoding is different: blendit ab condition: 000 hone on e cond iii I xxx DDDDDDDDD one on a one cal iii I xxx DDDDDDDDD one on a one cal become to be condition: 000 hone on a one cal iii I xxx DDDDDDDDD one on a one cal iii I xxx DDDDDDDDD one on a one cal iii I xxx DDDDDDDDD one on a one cal iii I xxx DDDDDDDDD one on a one cal iii I xxx DDDDDDDDD one on a one cal iii I xxx DDDDDDDDD one on a one cal iii I xxx DDDDDDDDDD one on a one cal iii I xxx DDDDDDDDDD one on a one cal iii I xxx DDDDDDDDDDDDD one on a one cal iii I xxx DDDDDDDDDDDDDDDDDDDDDDDDDDDDDD	ψ m	Th. Pa other wise, just go to next instruction
bcs 4MAIN go to 4MAIN if csi neoding is different: bicondi aD condition: 000 none cond III I XXX DDDDDDDDD OII Cc 100 Cs elates to Date 6 nearle: loop to substract 1 6836 times — to take times must ro. 4Caffoo add To. 8 Caff to 20 FFFF is 265536 to LDOP: Sub to. 21 To 4To -1	a br	#MRIN go so SMAIN IF 2=0
bcs 4MAIN go to 4MAIN if csi neoding is different: bicondi aD condition: 000 none cond III I XXX DDDDDDDDD OII Cc 100 Cs elates to Date 6 nearle: loop to substract 1 6836 times — to take times must ro. 4Caffoo add To. 8 Caff to 20 FFFF is 265536 to LDOP: Sub to. 21 To 4To -1		SMOTH OR TO MINISTRA OF C.C.O.
neoding is different: bicondi ad condition: 000 none cond III I XXX DDDDDDDDD lili I XXX DDDDDDDDD elates to Dart 6 lample: loop to substract I 65536 times — to take times must ro. \$Creft to 20x FFFF is 265536 to LDOP: Sub ro. \$1		
	b	SHARIN GO TO SHARIN IT CEL
	ncoding is diff	ent: bicondi #D condition: 000 none
011 CC 100 C3 eliates to Dart 6 usinple: loop to substract 1 68836 times — to take times must ro. \$0xFF00 add ro. \$0xFF resoxFFFF16.26553610 LDOP1 Sub ro. \$1 ro*70-1		001 eq
elates to part 6 Loop to substract 1 68836 times — to take times To . \$C_RFF.00 add To, \$C_RFF.00 LOOP 1 Sub To, \$1 To \$To -1	11 1 000	
100P		100 CS
mwt ro. # OxFFOO add ro. # OxFF ro. 20x FFFF 16 26553610 LOOP: Sub ro. 21 ro. 70 - 1	elates to pour 6	
mwt ro. # OxFFOO odd ro. # OxFF ro20xFFFF16 26553610 LOOP: Sub ro. % 1 ro % ro-1		
odd Ye. \$ Onff resource 16 26 555600	rample: loop to	ASTROCY 1 05556 FIRMES — TO Table TimeS
LDOP: Sub 70.81 70.470-1	met Y	40xFF00
LDOP: Sub 70.81 70.470-1	add n	8 Daff 10 = 00 FFFF 16 = 6 KE36 10
bina LOOP sohari vasuit = 0 , process to next step	LOOP: sub n	41 Yo 4 Yo - (
	bne t	D when result =0, process to next step