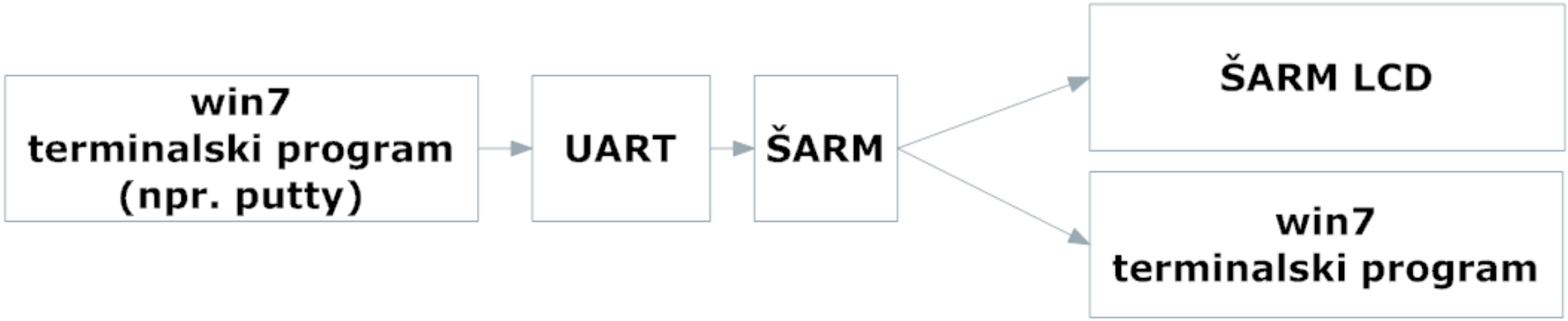
Vaja 11
Asinhroni serijski vmesnik (UART1)

1. **Napišite program, ki bo na asinhronih serijskih vratih sprejemal znak s terminalskega programa in jih prikazoval na LCD prikazovalniku.**
2. **Prejeti znak ESC (ascii koda 27) naj preklaplja med prikazom na sistemu ŠARM oz. V oknu terminalskega programa.**
3. **Program napišite še z uporabo prekinitev**



Preklop med prikazovanjem na LCD oz. na PC => prejeti znak ESC (ascii koda 27)

Asinhroni serijski sprejemnik / oddajnik (uart.c, uart.h)

UART1 inicializacija:
uart1_init(baud_rate,word_length, stop_bit, parity, parity_type, handshake, interrupts);

int baud_rate ... baud rate (bit/sec) – hitrost prenosa [9600,14400,...]

int word_length ... dolžina podatka [makro *word_length_5_bit, word_length_6_bit, word_length_7_bit* **ali** *word_length_8_bit***]**
int stop_bit ... št. stop bitov [makro *one_stop_bit* **ali** *two_stop_bits***]**

int parity ... preverjanje paritete [makro *disable_parity* **ali** *enable_parity***]**

int parity_type ... način preverjanja paritete [makro *odd_parity*, *even_parity*, *allways_one_parity* **ali** *allways_zero_parity***]**

int handshake ... inicializacija dodatnih pinov za strojni protokol prenosa;
0 => potrebujemo le pina p0.8 (txd) in p0.9 (rxd)
1 => dodatni pini so: p0.10 (rts), p0.11 (cts), p0.12 (dsr), p0.13 (dtr), p0.14 (dcd) in p0.15 (ri)

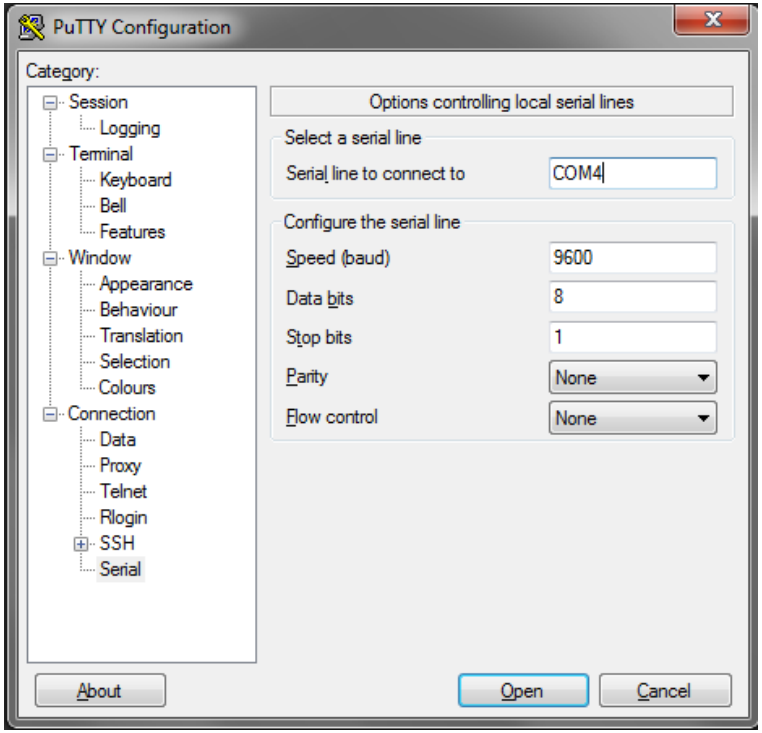
int interrupts ... omogočanje prekinitev [kombinacija 0 in zastavic *rx_data_available*, *thre_ier*, *rx_line_status* **and** *modem_status***]**

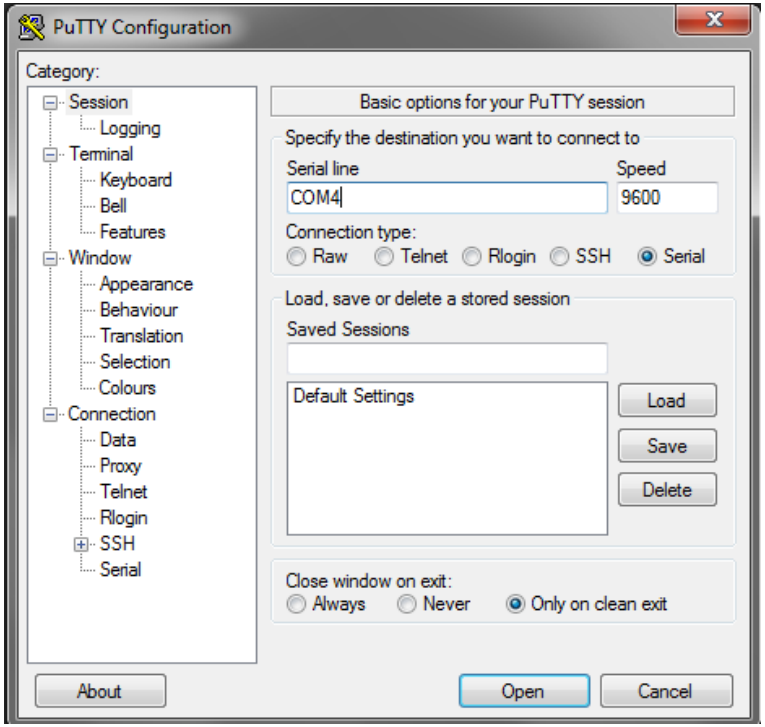
Asinhroni serijski sprejemnik / oddajnik (uart.c, uart.h)

Registri:

Name	Description	Bit functions and addresses								Access	Reset value[1]	Address
		MSB							LSB			
		BIT7	BIT6	BIT5	BIT4	BIT3	BIT2	BIT1	BIT0			
U1RBR	Receiver Buffer Register	8-bit Read Data								RO	NA	0xE001 0000 (DLAB=0)
U1THR	Transmit Holding Register	8-bit Write Data								WO	NA	0xE001 0000 (DLAB=0)
U1DLL	Divisor Latch LSB	8-bit Data								R/W	0x01	0xE001 0000 (DLAB=1)
U1DLM	Divisor Latch MSB	8-bit Data								R/W	0x00	0xE001 0004 (DLAB=1)
U1IER	Interrupt Enable Register	Reserved	Reserved	Reserved	Reserved	Enable Modem Status interrupt[2]	Enable RX Line Status Interrupt	Enable THRE Interrupt	Enable RX Data Available Interrupt	R/W	0x00	0xE001 0004 (DLAB=0)
U1IIR	Interrupt ID Register	FIFOs Enabled		Reserved	Reserved	IIR3	IIR2	IIR1	IIR0	RO	0x01	0xE001 0008
U1FCR	FIFO Control Register	RX Trigger		Reserved	Reserved	Reserved	TX FIFO Reset	RX FIFO Reset	FIFO Enable	WO	0x00	0xE001 0008
U1LCR	Line Control Register	DLAB	Set Break	Stick Parity	Even Parity Select	Parity Enable	Number of Stop Bits	Word Length Select		R/W	0x00	0xE001 000C
U1MCR[2]	Modem Control Register	Reserved	Reserved	Reserved	Loop Back	Reserved	Reserved	RTS	DTR	R/W	0x00	0xE001 0010
U1LSR	Line Status Register	RX FIFO Error	TEMT	THRE	BI	FE	PE	OE	DR	RO	0x60	0xE001 0014
U1MSR[2]	Modem Status Register	DCD	RI	DSR	CTS	Delta DCD	Trailing Edge RI	Delta DSR	Delta CTS	RO	0x00	0xE001 0018
U1SCR	Scratch Pad Register	8-bit Data								R/W	0x00	0xE001 001C
U1TER	Transmit Enable Register	TXEN	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	R/W	0x80	0xE001 0030

Terminalski program putty: konfiguracija





Izvedba z zanko:

start_up():
1. uart1_init(...)

2. Omogočanje oddajnika: postavitev bita 7 (maska *txen*) v reg. *U1TER*

main() – glavna zanka:
1. Preberi status linije (reg. *U1LSR*) => shranimo v spr. *status*

2. Če je znak na voljo => postavljen bit 0 (maska *rdr*) v spr. *status*

3. Podatek preberi iz reg. *U1RBR*

4. Če ni prišlo do napake pri prenosu => pobrisan bit 7 (maska *rxfe*) v spr. *status*

5. Če je prejeti znak ESC (ascii koda 27) => menjava izhoda

6. Izpis znaka na LCD oz. v oddajni reg. *U1THR*

Izvedba s prekinitvami:

start_up():

1. **Nastavimo VIC: prožili bomo uart1 prekinitve**

2. **Pri uart1_init(…) omogočimo *rx_data_available* prekinitev, ki se sproži:**
   - **ko se pojavi nov podatek**
   - **ko podatka predolgo nismo prebrali (timeout)**

uart1_ISR():

1. **Preberi status prekinitve (reg. *U1IIR*) => shranimo v spr. *status***

2. **Če je bila izdana nova prekinitev => pobrisan bit 0 (maska *interrupt_pending*) v *status***

3. **Vir prekinitve preberemo iz spr. *status* : biti 3-1 (maska *interrupt_id*)**

- 0x4 (makro *rx_data_available_id*) => sprejet je bil nov podatek
=> preberi znak (hkrati pobriše zahtevo po prekinitvi)
=> obdelaj znak (menjava izhoda, izpis na LCD/PC)

- 0xE (makro *character_time_out_id*) => timeout
=> preberi podatek (hkrati pobriše zahtevo po prekinitvi)

4. Ponastavi VIC