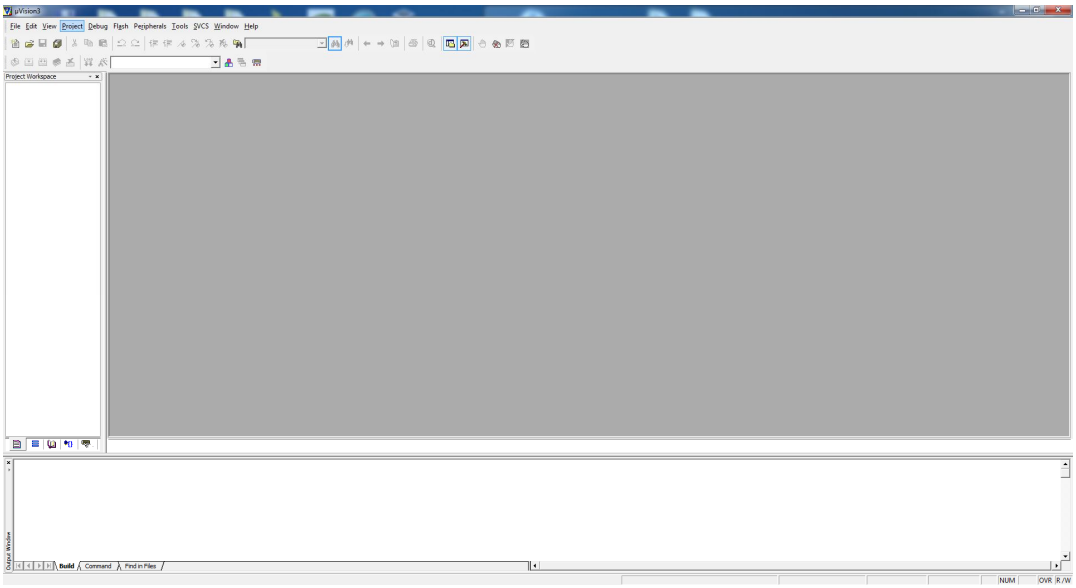
Vežba 5b

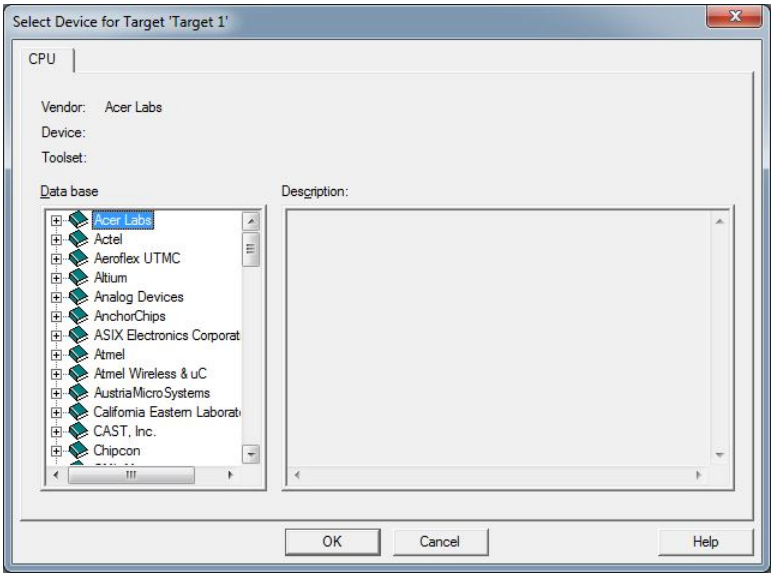
Upoznavanje sa Keil razvojnim okruženjem

U ovoj vežbi je cilj da se studenti upoznaju sa Keil razvojnim okruženjem. Razvojno okruženje se pokreće aktivacijom Uv3 prečice iz Hardware foldera na Desktop-u. U slučaju da nema foldera ili prečice, program je instaliran u C:\Keil\Uv3. Nakon pokretanja pojavljuje se glavni prozor koji izgleda kao na slici 5b.1.



Slika 5b.1. Izgled glavnog prozora radnog okruženja

Prvi korak je kreiranje novog projekta (Project -> New uVision project). Nakon odabira putanje i foldera pojavljuje se prozor za izbor mikrokontrolera kao na slici 5b.2. Izabrati Atmel AT89S51. **Ne treba dodavati *startup* kod**.



Slika 5b.2. Izbor mikrokontrolera

Dodati izvorni kod Primera 1 iz Vežbe 1. Kod se dodaje izborom opcije *Add files to Group 'Source Group 1'* desnim klikom na *Source Group 1*.

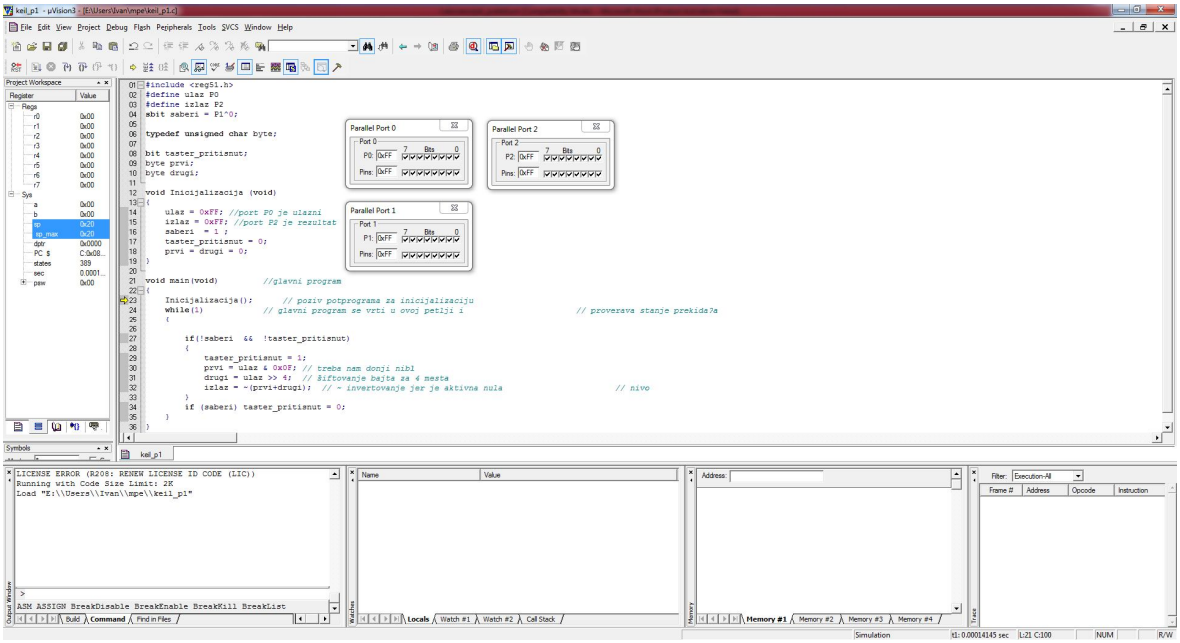
Nakon dodatog izvornog koda, izborom *Build target* (iz Project menija ili sa palete sa alatkama) kompajlirati program. Biće prijavljeno nekoliko grešaka koje su posledica razlika SDCC i Keil C51 kompajlera.

Za sada ispraviti sledeće:

1. 8051.h prebaciti u reg51.h
2. Umesto `#define saberi P1_0` staviti `sbit saberi = P1^0;`
3. Kod svih deklaracija koje su sa duplom donjom crtom (__) obrisati duple donje crte (npr. `__bit` treba da je `bit`)

Ostale specifičnosti Keil C51 kompajlera i jezika C će biti date u nastavku.

Ponoviti kompajliranje i ukoliko je prošlo bez grešaka pokrenuti *Debug* mod (*Debug* meni – *Start/stop debug session* ili izborom  na paleti sa alatkama). Iz *Peripherals* menija odabrati *I/O ports* i uključiti portove P0, P1 i P2. Nakon ovoga izgled radnog okruženja je dat na slici 5b.3.



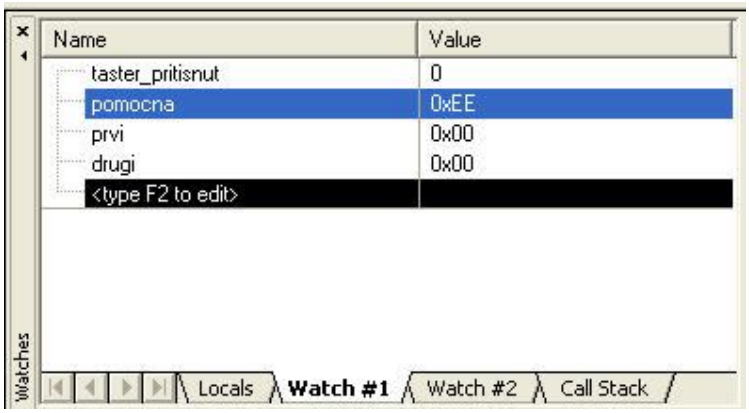
Slika 5b.3. Izgled radnog okruženja u *Debug* modu

Odabirom *Run* opcije iz *Debug* menija ili na paleti sa alatkama pokrenuti program, a potom ga protestirati izborom sabiraka na P0, emulacijom tastera pritiskom i otpuštanjem na P1.0 i praćenjem rezultata na P2 (videti sliku 5b.5). Emulaciju zaustaviti pokretanjem opcije *Halt*. Iz *Debug* moda se izlazi odabirom opcije *Start/stop debug session*.

Ukoliko postoje delovi programa koji nisu interesantni za posmatranje, možemo ih jednostavno preskočiti dodavanjem prekidnih tačaka (*breakpoint*). Na primer, nije nam interesantno da vidimo kako se izvršava funkcija inicijalizacije. Ali pre nego što se uvede prekidna tačka interesantno je pogledati kako se vrši inicijalizacija promenljivih. Za posmatranje promenljivih koristi se *Watch Window* koji se aktivira izborom opcije *Watch&Call Stack Window* iz menija *View*. Pojaviće se prozor u donjem desnom uglu radnog okruženja. U ovom prozoru se mogu posmatrati lokalne promenljive, stek, ali i promenljive koje smo mi definisali. Klikom na *Watch #1* jezičak, pojavljuje se novi prozor u kome se pritiskom na tipku F2 može uneti bilo koja definisana promenljiva. U toku simulacije od

posebne koristi je mogućnost menjanja vrednosti promenljivih čime se testiranje ubrzava i pojednostavljuje.

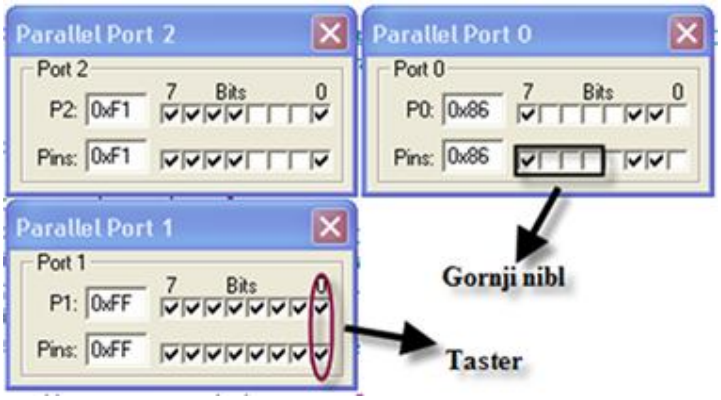
U našem primeru imamo 4 definisane promenljive i možemo ih sve posmatrati. Kada dođe do promene vrednosti, promenljiva se označi plavom bojom (sl. 5b.4).



Slika 5b.4. Watch prozor sa posmatranje promenljivih

Podesiti ulazne signale na portu P0. Na portu P2 posmatramo rezultat. Ukoliko želimo da saberemo donji i gornji nibl porta P0, potrebno je pritisnuti taster koji je vezan za nožicu P1.0, a to se takođe može odraditi u ovom meniju. Ukoliko želimo da pritisnemo taster, označićemo kućicu kao na slici (sl. 5b.5). Na slici je prikazan pritisnut taster.

Ako još napomenemo da se u *Peripherals* meniju mogu posmatrati i menjati još i prekidi, tajmeri, serijski port, onda posedujemo zadovoljavajući alat za emuliranje periferija.



Slika 5b.5. Rad sa portovima u Keil simulatoru

Keil proširenja standardnog C jezika za x51 mikrokontrolere

Keil C51 kompajler koristi brojne nove ključne reči koje ne postoje u standardnom C jeziku (sl. 5b.6). Ukrako ćemo objasniti značenje najviše korišćenih ključnih reči.

Arhitektura x51 podržava nekoliko fizički odvojenih memorijskih prostora i delova za smeštanje programa. Svaki memorijski prostor nudi određene prednosti i mane. Postoje memorijski prostori koji omogućavaju:

- Čitanje ali ne i upis
- I upis i čitanje
- I upis i čitanje, ali brže u odnosu na ostale memorije

_at_	far	sbit
alien	idata	sfr
bdata	interrupt	sfr16
bit	large	small
code	pdata	_task_
compact	_priority_	using
data	reentrant	xdata

Slika 5b.6. Ključne reči dodate ANSI C-u za lakši opis x51 arhitekture

Da bi se mogli dodati delovi koda u asembleru (*inline*) potrebno je koristiti blok naredbi ASM/ENDASM na primer:

```
#pragma asm
  nop;
#pragma endasm
```

Pored toga je potrebno uključiti *Generate Assembler SRC File* i *Assemble SRC File* u *Properties* tabu od *Options for File* izvornog fajla projekta.

Programska (*code*) memorija ima samo mogućnost čitanja i može biti po realizaciji unutrašnja, spoljašnja ili kombinacija ove dve memorije. Programskoj memoriji se može pristupiti pomoću memorijskog tipa **code** definisanog u okviru C51 standardne biblioteke.

Unutrašnja memorija za podatke se nalazi na samom čipu i u nju se može i upisivati i čitati. Postoje razne varijante čipova u zavisnosti od količine unutrašnje memorije za podatke, ali u našem slučaju radićemo sa varijantama koje sadrže 256k unutrašnje memorije za podatke. Memoriji za podatke se brzo pristupa jer se koristi 8-bitna adresa. Za pristup internoj memoriji se koriste tri različita memorijska tipa:

- **data** - označava da se pristupa internoj memoriji podataka uz direktno adresiranje, što omogućava brz pristup (128B).
- **idata** - označava da se pristupa celoj memoriji za podatke (256B) uz indirektno adresiranje.
- **bdata** – označava da se pristupa lokacijama ukupne veličine od 16 bajtova koje se mogu adresirati po bitovima.

Za razliku od unutrašnje, spoljašnja memorija za podatke je sporija, jer se pristup vrši preko pokazivača na podatke. Maksimalno je podržano 64k spoljašnje memorije za podatke. Vrlo je važno da se shvati da se ovaj adresni prostor ne mora koristiti samo za pristupanje memoriji. Naš hardverski dizajn može mapirati određene periferne sklopove unutar memorijskog prostora što će biti tema naredne vežbe. C51 kompajler nudi dva memorijska tipa za pristup spoljašnjoj memoriji za podatke: