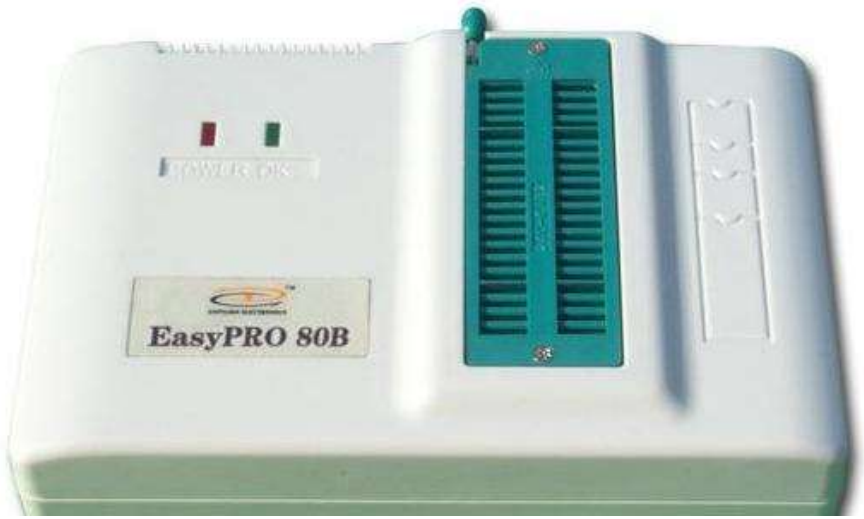
Procesorski sistemi v telekomunikacijah
Nalaganje in razhroščevanje programov

Zapis programa s programatorjem

- Program v (E)EPROM/FLASH pomnilniku v ločenem čipu
- Čip odstranimo iz sistema in vanj s pomočjo programatorja zapišemo program
- Običajno je treba program pretvoriti iz oblike, ki jo dobimo iz povezovalnika v obliko, ki jo pozna programator
- Formati: Intel HEX, Motorola S (S19), ... od leta 1970 naprej
- Pogosto je FLASH pomnilnik v istem čipu kot procesor... Procesor vstavimo v programat in vanj zapišemo program



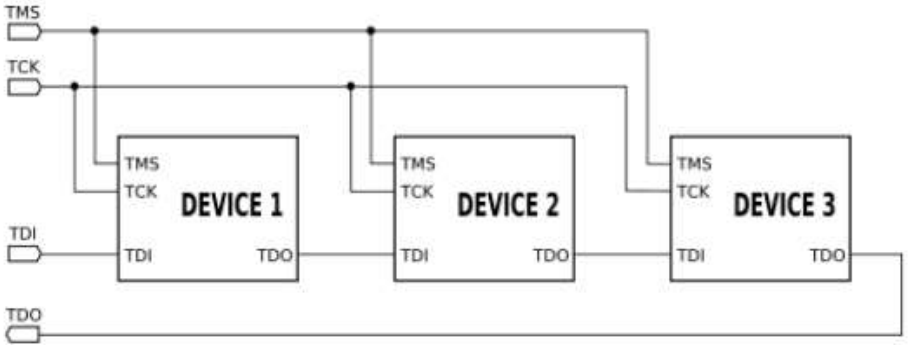
Programiranje med delovanjem (In System Programming, ISP)

- Nekateri mikroprocesorji z vgrajenim FLASH pomnilnikom omogočajo, da v njih prenesemo program, brez da bi jih odstranili iz vezja **(In System Programming, ISP)**
- Najbolj pogosta načina nalaganja programa sta serijska povezava in vmesnik JTAG

- Primer: v LPC2138 lahko po reset-u (če je P0.14 na nizkem nivoju) naložimo program preko vmesnika UART0 (P0.0, P0.1 = TxD0, RxD90) v FLASH pomnilnik s hitrostmi do 230400b/s (23kB/s)

Vmesnik JTAG

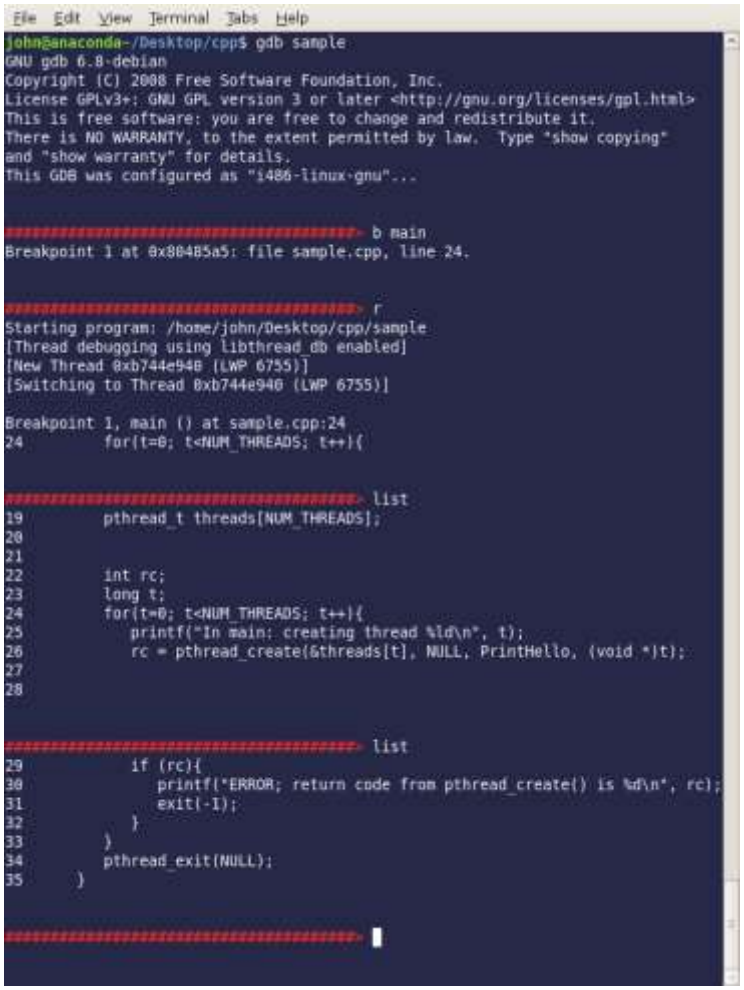
- JTAG (Joint Test Action Group, IEEE 1149.1)
- Podoben vodilu SPI, 4 signali:
  TMS (test mode select) – enakovreden SS (slave select) signalu SPI
  TCK (test clock) – enakovreden SCLK signalu SPI
  TDI (test data input) – enakovreden MOSI signalu SPI
  TDO (test data output) – enakovreden MISO signalu SPI
- Verižna vezava naprav, ki jih krmilimo preko JTAG povezave
  Pogosto imamo le eno napravo v verigi, t.j. mikroprocesor
- Omogoča krmiljenje in odčitavanje stanja naprav v verigi.
- Tipične hitrosti od 10Mb/s do 100Mb/s
- Primer: JTAG v LPC2138 P1.27-P1.30 ... TDO, TDI, TCK, TMS
- Standard ne določa ukazov, ki se
  prenašajo preko povezave JTAG
  Vsak proizvajalec ima svoje ukaz



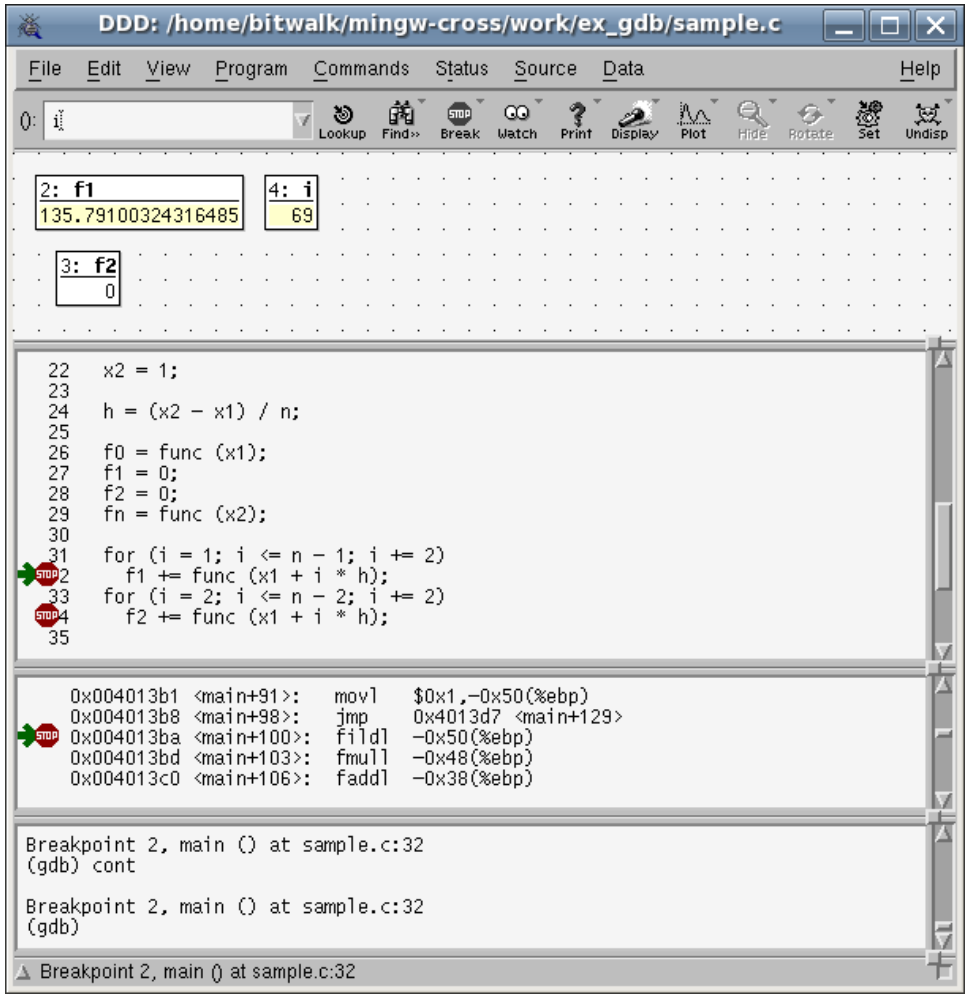
Razhroščevanje (debugging)

- Postopek iskanja napak v programu.
- **Preprost pristop**: takoimenovano **printf razhroščevanje**
  V program dodamo stavke printf, ki redno izpisujejo vrednosti spremenljiv in stanje programa. Izpisana sporočila beremo preko serijske povezave ali pa na prikazovalniku (pogosto je ta del mikroprocesorskega sistema).
- V skrajnih primerih namesto tekstovnih sporočil uporabljamo lučke ali pa kar signale, ki jih opazujemo na osciloskopu.
- Pristop je preveč preprost, da bi z njim lahko odkrili večje napake.
- **Posebni programi in strojna oprema za razhroščevanje – razhroščevalnik (debugger)**
  Z njihovo pomoéjo lahko
  - ustavimo program na določenih mestih (prekinitvene točke, breakpoints),
  - gremo skozi program po korakih (korakanje, stepping),
  - pregledujemo vrednosti spremenljivk, registrov in pomnilnika (watch)

Razhroščevalnik (Debugger)
Primer: GDB



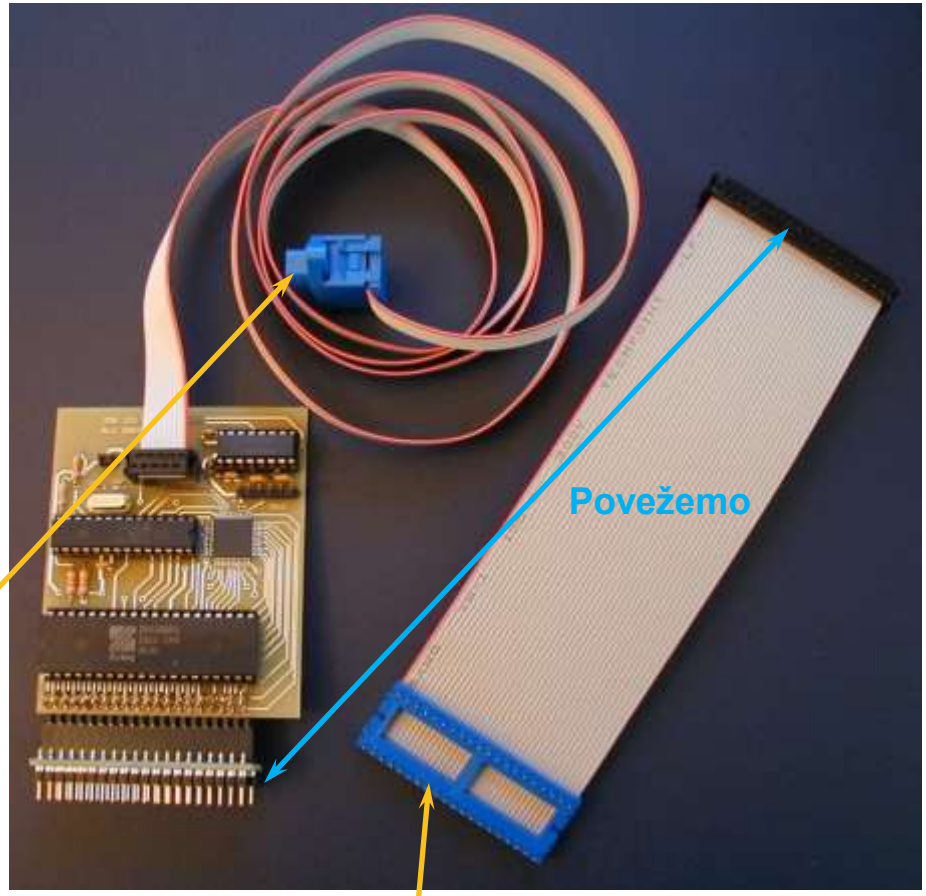
Konzolni način dela v GDB



Grafični uporabniški vmesnik (GUI) za GDB

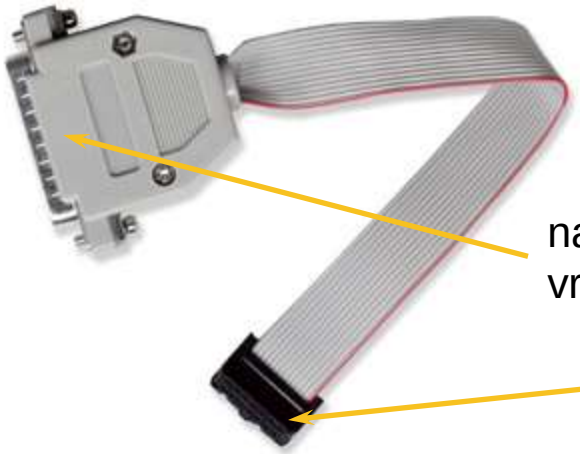
Strojna podpora za razhroščevanje Emulatorji

- Emulator (In Circuit Emulator, ICE) je vezje, ki nadomesti mikroprocesor.
- Vstavimo ga v mikroprocesorski sistem namesto mikroprocesorja.
- Hkrati ICE priklopimo še na osebni računalnik, kjer s pomočjo ustrezne programske opreme (pripadajočega razhroščevalnika) iščemo napake v programu.
- Na sliki je emulator Z80 – Z80 ICE
- Relativno visoka cena.



Mikroprocesorji z vgrajenim emulatorjem

- Večina današnjih mikroprocesorjev.
- Komunikacija z emulatorjem poteka ponavadi preko vmesnika JTAG.
- Za povezavo z osebnim računalnikom skrbi vmesniško vezje.
- Programska oprema za razhroščevanje mora znati komunicirati z JTAG vmesnikom in preko njega z emulatorjem v mikroprocesorju.
- Pri okolju WinIDEA (š-ARM) je vmesnik USB/JTAG vgrajen v ploščico š-ARM.



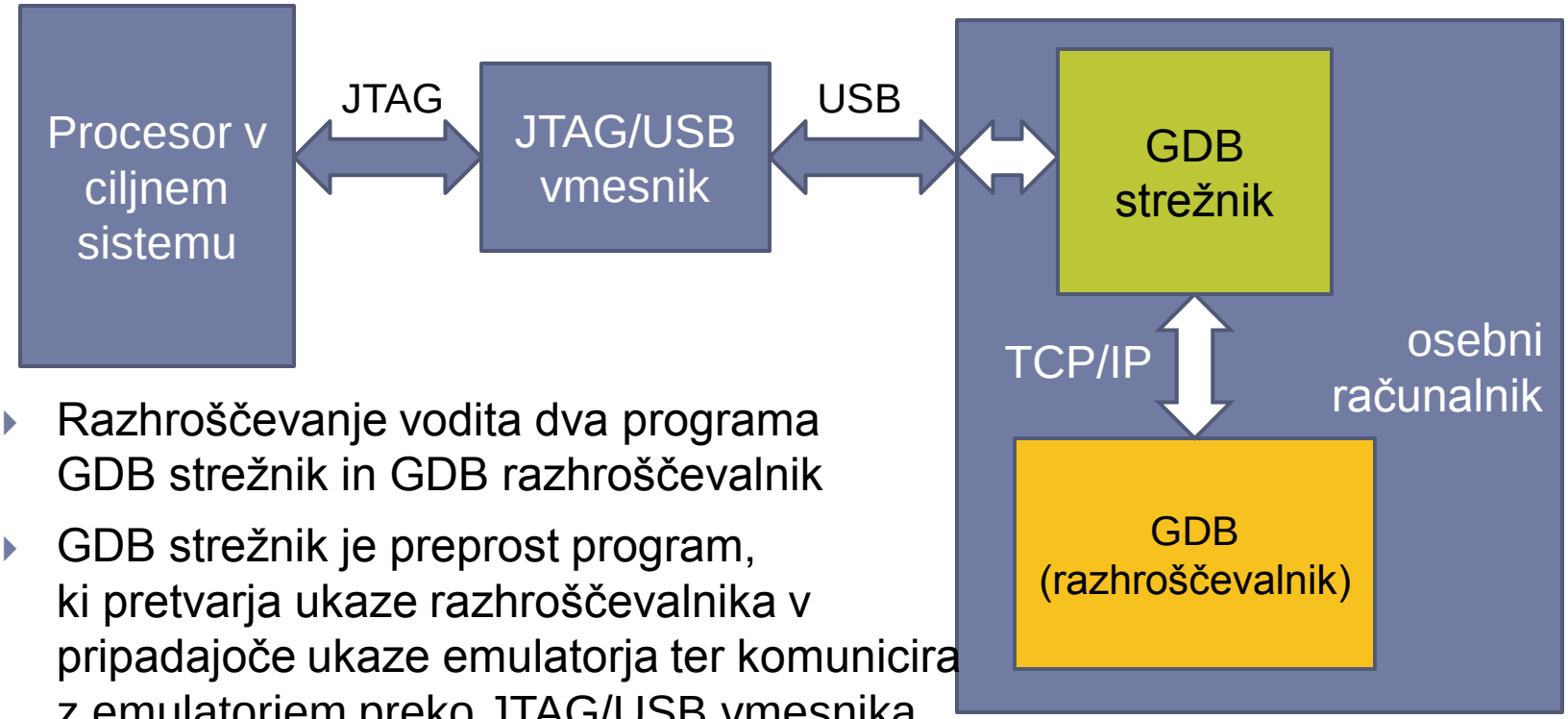
Vmesnik JTAG-paralelna vrata
za procesorje ARM (Olimex)

Vmesnik JTAG-USB
za procesorje ARM (Olimex)

Priprava na razhroščevanje

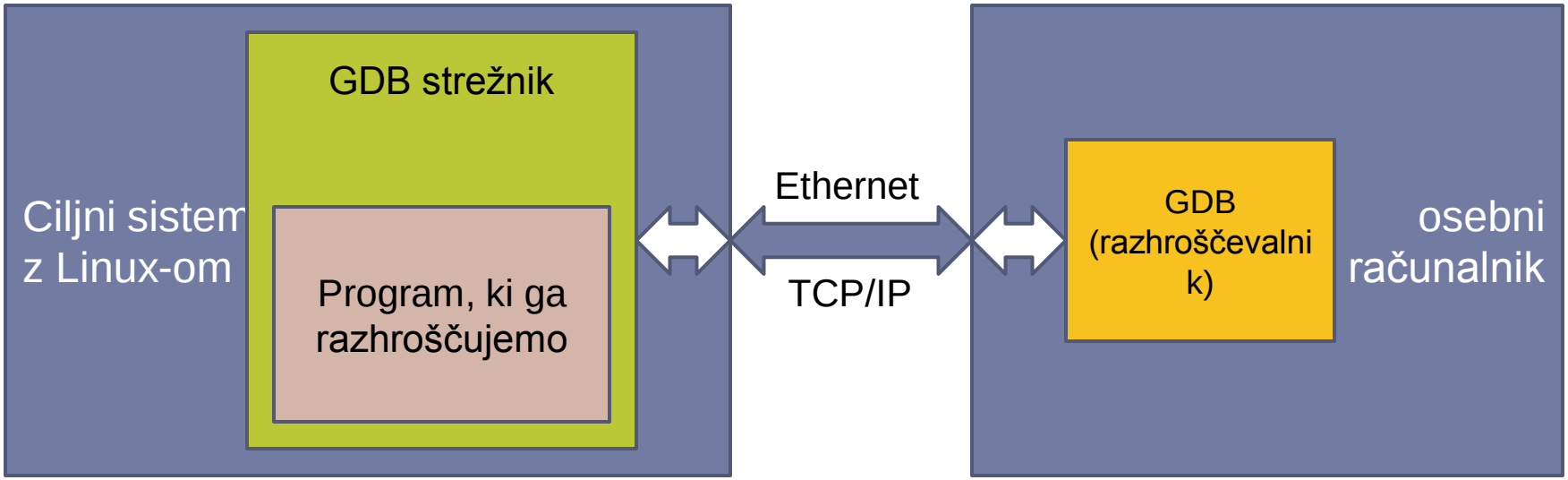
- Program je potrebno prevesti v posebno obliko za razhroščevanje, ki vsebuje pomožne informacije (debug information) o spremenljivkah in stavkih (npr. kje v pomnilniku se hranijo).
- Pomožne informacije so lahko del objektnih datotek in končnega programa (npr. pri gcc) ali pa se hranijo v ločenih datotekah (npr. Microsoftov prevajalnik za C).
- Program je treba zagnati s pomočjo razhroščevalnika, ki takoj po zagonu prevzame nadzor nad mikroprocesorjem.
- Razhroščevalnik ponavadi teče na osebnem računalniku, ki je preko vodila USB in vmesnika USB/JTAG povezan s ciljnim mikroprocesorjem, ki

Zgradba sistema za razhroščevanje na osnovi GNU orodij gcc/gdb



- Razhroščevanje vodita dva programa GDB strežnik in GDB razhroščevalnik
- GDB strežnik je preprost program, ki pretvarja ukaze razhroščevalnika v pripadajoče ukaze emulatorja ter komunicira z emulatorjem preko JTAG/USB vmesnika
- TCP/IP komunikacija med GDB strežnikom in GDB razhroščevalnikom.
- Uporabnik (programer) upravlja razhroščevanje preko GDB razhroščevalnika.
- TCP/IP omogoča, da GDB strežnik in GDB razhroščevalnik tečeta na dveh ločenih računalnikih, ki sta povezana preko interneta.

Ciljni sistemi z operacijskim sistemom Primer: Linux/gcc/gdb

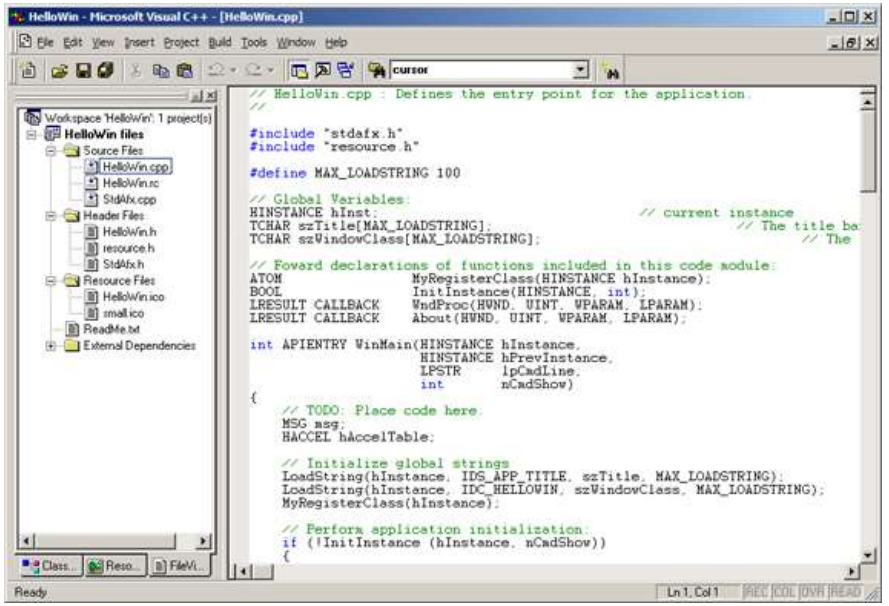


- Program mora biti naložen na ciljni sistem, zaženemo GDB strežnik.
- Ob zagonu mu podamo program, ki ga bomo razhroščevali.
- Na osebnem računalniku zaženemo GDB razhroščevalnik.
- GDB razhroščevalnik se poveže z GDB strežnikom preko mrežne povezave.
- TCP/IP komunikacija med GDB strežnikom in GDB razhroščevalnikom.
- Razhroščevanje programa poteka s podporo operacijskega sistema. JTAG vmesnik in emulator nista potrebna.

Integrirano razvojno okolje
Integrated Development environment (IDE)

- Na enem mestu zbrana orodja za pisanje, prevajanje in razhroščevanje.
- Upravljanje s projekti (programi, ki so sestavljeni iz večih datotek).
- Preprosto razhroščevanje.
- WinIDEA je IDE za š-ARM.

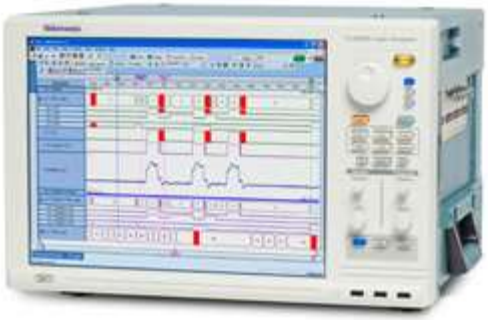
Okolje ECLIPSE



Okolje Microsoft Visual Studio

Razhroščevanje na nivoju signalov
Logični analizator

- Podoben osciloskopu, snema digitalne signale od trenutka proženja.
- Omogoča iskanje napak v signalih (glitch, motnje, odboji, ..)
- Ločljivost cca. 20ps (50GHz), nekaj 10 do več kot 100 kanalov
- Posnetki dolgi več 100MB
- Razumejo protokole, kot je naprimer PCI Express, ...



Sled programa (Trace)

- Dnevnik dogajanja v programu
- Vrednosti izbranih spremenljivk opazujemo v sledilnih točkah (tracepoint)

```
i=2;
p=1;
// sledila točka 1
while (i<=6) {
  p=p*i;
  // sledila točka 2
  i=i+1;
  // sledila točka 3
}
// sledila točka 4
```

sledilna točka	i	p
1	2	1
2	2	2
3	3	2
2	3	6
3	4	6
2	4	24
3	5	24
2	5	120
3	6	120
2	6	720
4	7	720