Arhitekture i algoritmi
DSP 2

Auditorne vežbe AU-8[0]
Model 3

Odsek za računarsku tehniku i računarske komunikacije

Model 3

- Prebacivanje na CLIDE/CS497xx okruženje
- Umesto emulacionih fajlova (stdfix_emu i fixed_point_math) koristi se ugrađena biblioteka **stdfix**
- Integracija modela 2 u CLIDE Standalone projekat
  - Čisto C rešenje (minimum za prolaz na projektu)
  - Funkcija obrade u ASM-u (mešavina C i ASM)

Čisto C rešenje

- Ispraviti sve C++ „funkcionalnosti“ kao što su definisanje lokalnih promenljivih u proizvoljnim delovima koda (sve ih pomeriti na početak funkcije)
- U common.h treba da se nalazi deo koda iz stdfix_emu.h sa definicijama FRACT_NUM i long_accum tipova
- Srediti izlazni broj kanala u main funkciji

```
#if defined(__CCC)

#include <stdfix.h>

#define FRACT_NUM(x) (x##r)
#define LONG_FRACT_NUM(x) (x##lr)
#define ACCUM_NUM(x) (x##lk)

#define FRACT_NUM_HEX(x) (x##r)

#define FRACT_TO_INT_BIT_CONV(x) (bitsr(x))
#define INT_TO_FRACT_BIT_CONV(x) (rbits(x))

#define long_accum long accum
#define long_fract long fract

#endif
```

Čisto C rešenje kao prelaz za ASM

- Dodati __memY kvalifikator kod sampleBuffer-a (ako se ne definiše kvalifikator ide u X memoriju, a IOBuffer-i se nalaze u Y memoriji)

```
// IO Buffers
__memY DSPfract sampleBuffer[MAX_NUM_CHANNEL][BLOCK_SIZE];
```

- Izmeniti sve pokazivače koji pokazuju na sampleBuffer (posredno i neposredno) da pokazuju na __memY*

```
// get address of the first sample in the current channel
__memY DSPfract* samplePtrIn = *(pIn + i);
__memY DSPfract* samplePtrOut = *(pOut + i);
__memY DSPfract* centerSamplePtr = *(pOut + CENTER_CH);
```

Funkcija obrade u ASM-u

Simboli definisani u asemblerskom jeziku:

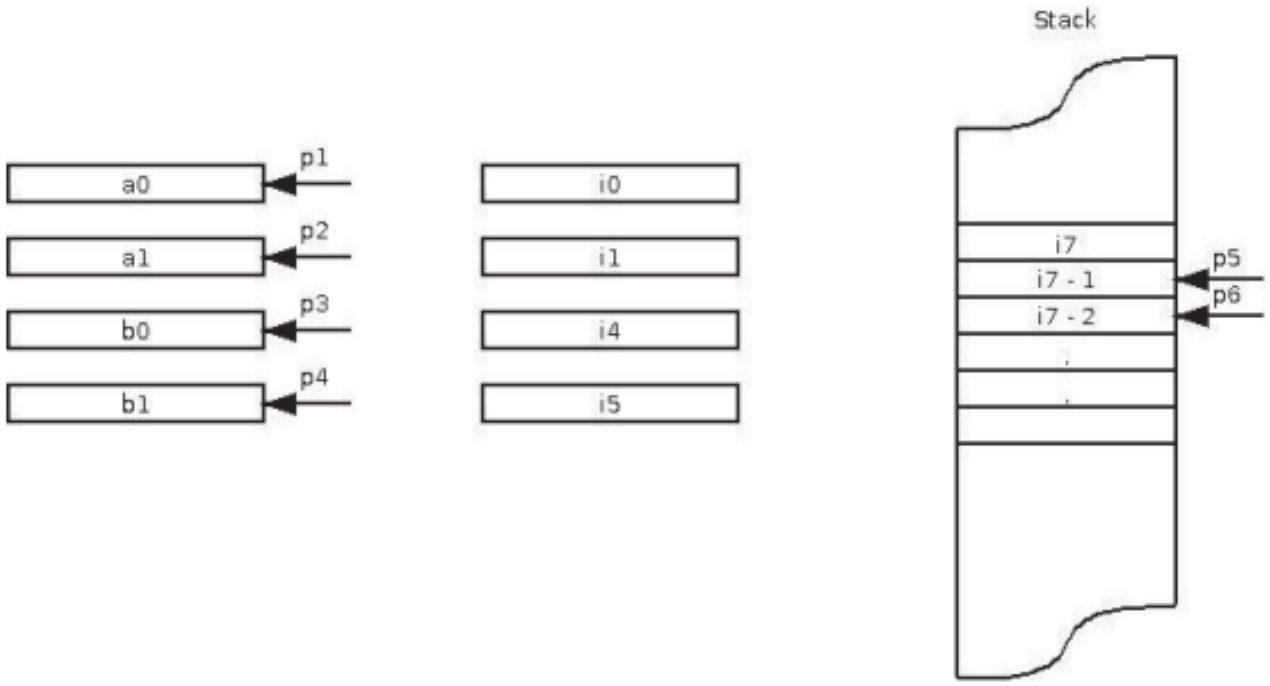
```
    .public _processingASM
    .public _y

    .xdata_ovly

_y    .dw (0)

    .code_ovly
_processingASM:
    a0=xmem[_y]
    a0 = a0 >> 1
    xmem[_y]=a0
    ret
```

Pristup simbolima iz programskog jezika C:

```
extern void processingASM();
extern __memX int y;
void main()
{
    y++;
    processingASM();
}
```

Konvencija CCC2 poziva

```
void foo(int p1, int p2, int p3, int p4, int p5, int p6)
```



Slika 6.1 - Ilustracija prosleđivanja parametara kod funkcije foo

6.2.2.2 Pozivna konvencija kod CCC2 programskog prevodioca