Basic Program Structure (Asm + C)

Standard form Asm

label: opcode operands comment

labels name current position in code

current pos. is addr at which next

inst. goes
for:  add $1, %eax  
  sbb $1, %ebx  

bne foo  ; br. to foo if zero not set  

directives are aimed at the assembler  

.equ MAX_CNT, 12  
.movl $MAX_CNT, %ebx  
.ppseud $12, %ebx  

MAX_CNT .eqn 12
.text

.tells assembler we are in text (code) section

Other common section names include

.ro data
.read only data

.data
.initialized data

.bss
.uninitialized data

comes from IBM 704 assembly lang. late 1950s

"Block storage start"
distinct sections useful in embedded systems

alternative syntax

section name

labels important in data section

section .data

i: .int 0
str: .string "this is an example"
var: .int 3
is label that means "right here"

exp makes $+4$ means $4$ adds beyond current position

add

.org $+8$

sub

$xx\ xx\ xx\ xx$ (8 bytes empty)

$xx\ xx\ xx\ xx\ xx$
/*
 * initial set of comments describing
 * func. of prog. or file
 */

.text

.global main

main:
push ebp
movl ebp, esp
pushl %ebx
pushl %esi
pushl %edi

; assembly code to define function */
C program structure

classically <sep> into two sets of files
   — c program files
   — h header files

/*
 * bar.h
 */
/* global comments */
#define MAX_CNT 12 // preprocessor dir
/ * bar.c
 * header comment
 */

#include <stdio.h>  // system
#include "bar.h"  // local

int var = 1;  // global var
int x;  // local var

main (...) {
    int var = 2;  // local var
    /* code */
    x = var;
}