Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

Instructions:

Total: **40 marks**

1. This question paper contains a total of 9 pages (9 sides of paper). Please verify.
2. Write your name, roll number, department, section on **every side of every sheet** of this booklet
3. Write final answers **neatly** in the given boxes.

Problem 1. (10 points) Provide the state-space representation of the dynamics of a DC Motor. Assume that there is no additional load on the motor. Next, Design a Simulink model to capture the dynamics and simulate the model for an input PWM voltage signal with magnitude 1V, frequency 1 kHz and duty cycle 0.1. Assume that the kinetic friction of the motor is negligible. Take the values of the other parameters from Example 7.13 in [LS15].

[LS15] Edward A. Lee and Sanjit A. Seshia, Introduction to Embedded Systems, A Cyber-Physical Systems Approach, Second Edition, http://LeeSeshia.org, ISBN 978-1-312-42740-2, 2015.

Answer

- The dynamics of the DC motor can be jointly defined by following 2 equations

$$Idw(t)/dt = K_t i(t) - \eta w(t) - \tau(t) \tag{1}$$

$$v(t) - R_i(t) - Ldi(t)/dt = k_b w(t) \tag{2}$$

- Rearranging terms in equation 1 and 2, we will get

$$(1/L)(v(t) - R_i(t) - k_b w(t)) = di(t)/dt \tag{3}$$

$$(1/k_t)(Idw(t)/dt + \eta w(t) + \tau(t)) = i(t) \tag{4}$$

- Differentiating equation 4 and equating it will 3 will give

$$(1/L)(v(t) - Ri(t) - k_b w(t)) = 1/K_t(d^2w(t)/dt^2) \tag{5}$$

- Now substituting the value of i(t) from equation (4) and using the fact that load τ is negligible and kinetic friction, η is negligible and rearranging the terms, we have

$$d^2w(t)/dt^2 = (K_t/IL)v(t) - (R/L)dw(t)/dt - (k_t k_b/IL)w(t) \tag{6}$$

- The constants are used as described in the exercise. Then
- $(K_t/IL) = 5.9 * 10^{-3}/(3.88 * 10^{-7} * 1.1 * 10^{-4}) = 138238000$
- $R/L = 1.71/(1.1 * 10^{-}4) = 15545.45$
- $(k_t k_b/IL) = (5.9 * 10^{-3} * 2.75 * 10^{-4})/(3.88 * 10^{-7} * 1.1 * 10^{-4}) = 38105.46$

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

Below is the Simulink implementation of the dynamics of Dc motor based on the differential equation above The above plot if the angular velocity plot and it can be concluded that the angular velocity

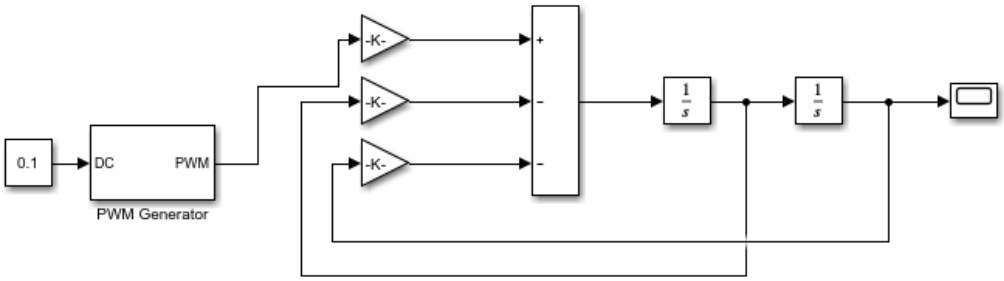


Figure 1: *
DC Motor

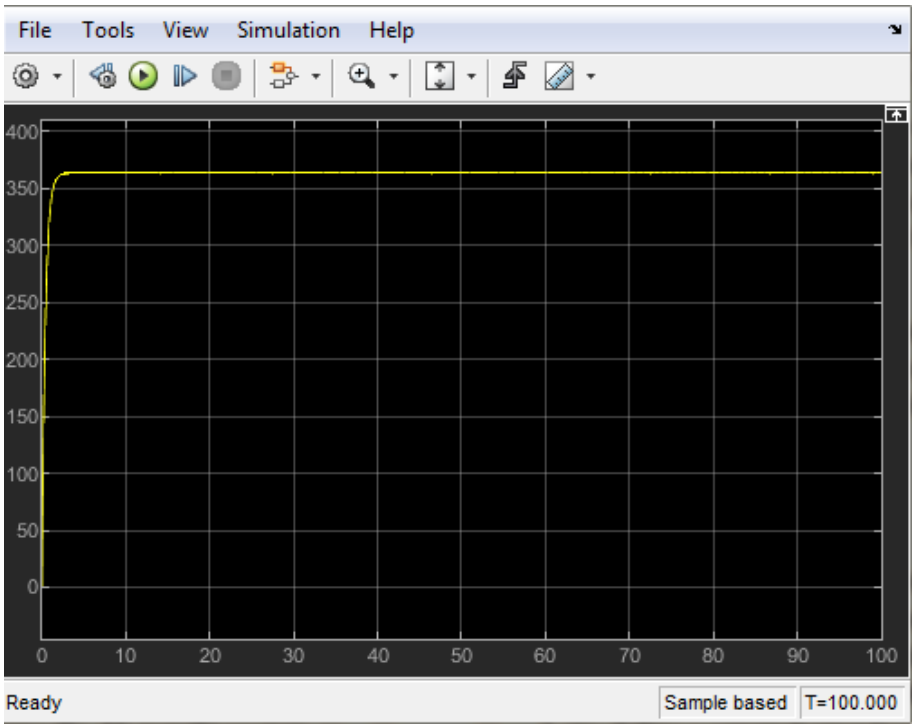


Figure 2: *
Angular velocity Plot

has attained a steady state at around 370 RPM.

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

Problem 2. (20 points) Consider the vehicle steering control problem in Example 6.4 in [AM09]. Assume that k1 = 1, k2 = 1.6 and kr = 1. Model the control system in Simulink using double precision floating point arithmetic. Now replace the model of the controller with the ones that use 16 bit and 8-bit fixed-point arithmetic. In each case, determine the fixed-point data types precisely. Plot the difference between the first state for the floating point controller and that for the fixed point controllers. Generate code for both the floating point controller and the fixed-point controllers using different optimization options. Describe your experience with code generation.
[AM09] K. J. Astrom and R. M. Murray. Feedback Systems: An Introduction for Scientists and Engineers. Princeton University Press, 2009.
http://www.cds.caltech.edu/∼murray/books/AM05/pdf/am08-complete_22Feb09.pdf.

Solution

Below diagram represents the Simulink implementation of Controller for Vehicle steering system with K, K_r used as described.

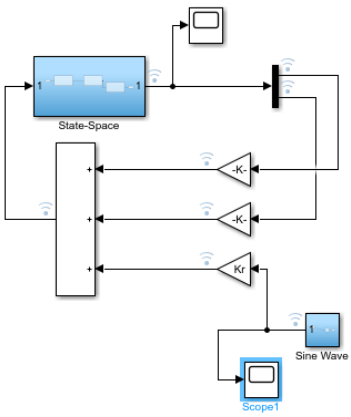
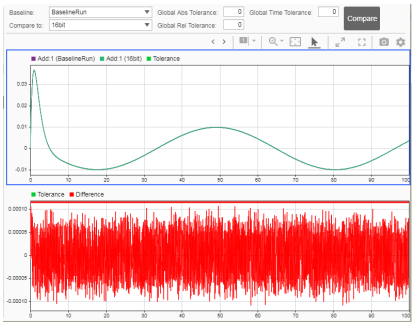
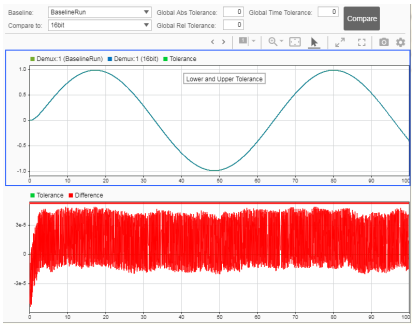


Figure 3: *
Controller for Vehicle Steering System

Comparison of floating point with 16 bit arithmetic

Below plots shows the comparison of the outputs along all the signal lines in the controller implementation shown above.





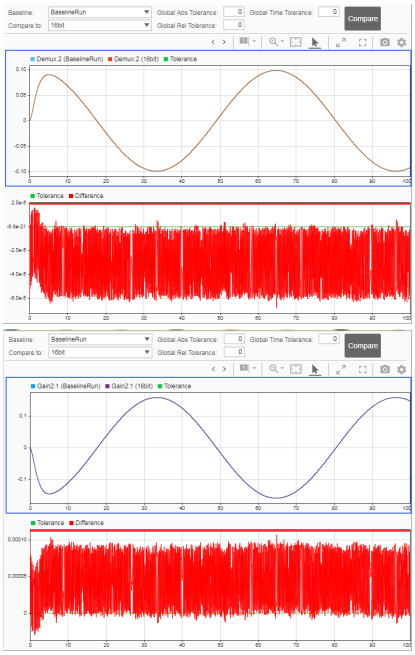
Name: Ajita Shree

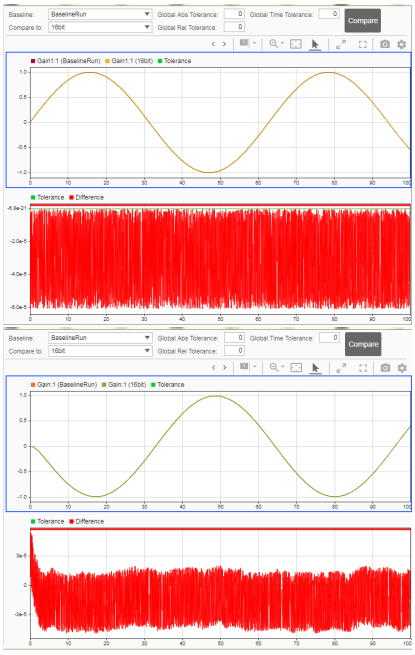
Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

The above part of the plot shows the signal output for while using floating point arithmetic vs 16 bit fixed point arithmetic whereas the below part of the plot shows the difference between the two.





The fixed point data type in this case would be fixdt(1, 16, 14) for output of the Gain 1, 3 block, fixdt(1, 16, 17) for Gain 2 block and fixdt(1, 16, 19) for the output of the Add block

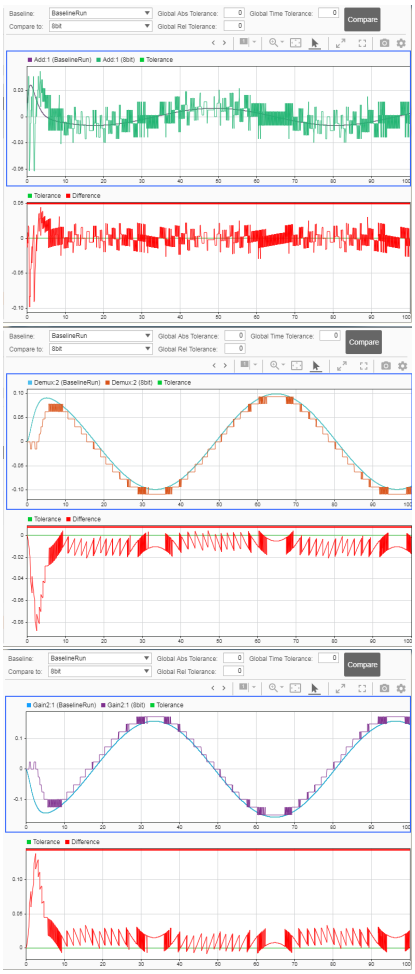
Name: Ajita Shree

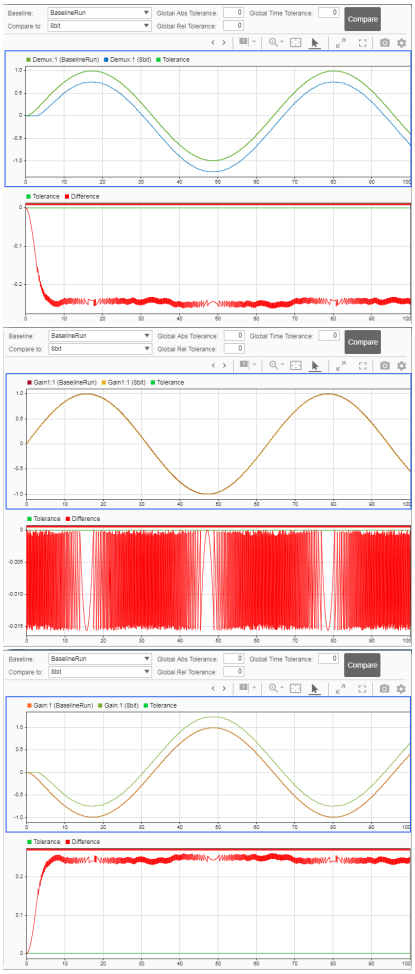
Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

Comparison of floating point with 8 bit arithmetic





We can observe here that the difference is more compared to the difference observed with the 16-bit fixed point arithmetic.

The fixed point data type in this case would be fixdt(1, 8, 6) for output of the Gain 1, 3 block, fixdt(1, 8, 9) for Gain 2 block and fixdt(1, 8, 11) for the output of the Add block.

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

Observations

- The above comparison plots are based on the fixed point data types proposed by Fixed Point tool for both 16 as well 8 bit.
- The comparison of results between floating point and 16-bit arithmetic is closer compared to floating point and 8-bit arithmetic as visible from the above graphs
- The Code has been generated for both the cases using **Simulink Coder** and **Embedded Coder**. All the .h and .c files have not been included here to keep the description precise.
- Different optimization techniques have been tried from the settings tab i.e. **Non-adaptive and Adaptive** etc
- The experience of code generation process was very good because the code was generated very quickly. Also, we have the flexibility to choose between algorithms and other features such as whether we want our build time to quick or run time to quick etc. The generated code snippet is as follows.

```
#include "State.h"
#ifndef rtmIsMajorTimeStep
# define rtmIsMajorTimeStep(rtm)       (((rtm)->Timing.simTimeStep) == MAJOR_TIME_STEP)
#endif

#ifndef rtmIsMinorTimeStep
# define rtmIsMinorTimeStep(rtm)       (((rtm)->Timing.simTimeStep) == MINOR_TIME_STEP)
#endif

#ifndef rtmSetTPtr
# define rtmSetTPtr(rtm, val)          ((rtm)->Timing.t = (val))
#endif

#ifndef UCHAR_MAX
#include <limits.h>
#endif

#if ( UCHAR_MAX != (0xFFU) ) || ( SCHAR_MAX != (0x7F) )
#error Code was generated for compiler with different sized uchar/char. \
#endif

#if ( USHRT_MAX != (0xFFFFU) ) || ( SHRT_MAX != (0x7FFF) )
#error Code was generated for compiler with different sized ushort/short. \

#endif

#if ( UINT_MAX != (0xFFFFFFFFU) ) || ( INT_MAX != (0x7FFFFFFF) )
#error Code was generated for compiler with different sized uint/int. \

#endif

#if ( ULONG_MAX != (0xFFFFFFFFU) ) || ( LONG_MAX != (0x7FFFFFFF) )
#error Code was generated for compiler with different sized ulong/long. \

#endif
```

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

```
RT_MODEL *const rtM = &rtM_;
extern void State_derivatives(void);
static void rt_ertODEUpdateContinuousStates(RTWSolverInfo *si )
{
  static const real_T rt_ODE3_A[3] = {
    1.0/2.0, 3.0/4.0, 1.0
  };

  static const real_T rt_ODE3_B[3][3] = {
    { 1.0/2.0, 0.0, 0.0 },

    { 0.0, 3.0/4.0, 0.0 },

    { 2.0/9.0, 1.0/3.0, 4.0/9.0 }
  };

  time_T t = rtsiGetT(si);
  time_T tnew = rtsiGetSolverStopTime(si);
  time_T h = rtsiGetStepSize(si);
  real_T *x = rtsiGetContStates(si);
  ODE3_IntgData *id = (ODE3_IntgData *)rtsiGetSolverData(si);
  real_T *y = id->y;
  real_T *f0 = id->f[0];
  real_T *f1 = id->f[1];
  real_T *f2 = id->f[2];
  real_T hB[3];
  int_T i;
  int_T nXc = 2;
  rtsiSetSimTimeStep(si,MINOR_TIME_STEP);

  (void) memcpy(y, x,
                (uint_T)nXc*sizeof(real_T));
  rtsiSetdX(si, f0);
  State_derivatives();

  hB[0] = h * rt_ODE3_B[0][0];
  for (i = 0; i < nXc; i++) {
    x[i] = y[i] + (f0[i]*hB[0]);
  }
```

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering


```
  rtsiSetT(si, t + h*rt_ODE3_A[0]);
  rtsiSetdX(si, f1);
  State_step();
  State_derivatives();

  for (i = 0; i <= 1; i++) {
    hB[i] = h * rt_ODE3_B[1][i];
  }

  for (i = 0; i < nXc; i++) {
    x[i] = y[i] + (f0[i]*hB[0] + f1[i]*hB[1]);
  }

  rtsiSetT(si, t + h*rt_ODE3_A[1]);
  rtsiSetdX(si, f2);
  State_step();
  State_derivatives();
  for (i = 0; i <= 2; i++) {
    hB[i] = h * rt_ODE3_B[2][i];
  }

  for (i = 0; i < nXc; i++) {
    x[i] = y[i] + (f0[i]*hB[0] + f1[i]*hB[1] + f2[i]*hB[2]);
  }

  rtsiSetT(si, tnew);
  rtsiSetSimTimeStep(si,MAJOR_TIME_STEP);
}

    void State_step(void)
{
  if (rtmIsMajorTimeStep(rtM)) {
    rtsiSetSolverStopTime(&rtM->solverInfo,((rtM->Timing.clockTick0+1)*
      rtM->Timing.stepSize0));
  }

  if (rtmIsMinorTimeStep(rtM)) {
    rtM->Timing.t[0] = rtsiGetT(&rtM->solverInfo);
  }
  rtY.Out1[0] = (int8_T)floor(rtX.ReplicaOfSource_CSTATE[0] * 64.0);
  rtY.Out1[1] = (int8_T)floor(rtX.ReplicaOfSource_CSTATE[1] * 64.0);
  rtDW.DTC_input_1 = (real_T)rtU.In1 * 0.00048828125;
  if (rtmIsMajorTimeStep(rtM)) {
    rt_ertODEUpdateContinuousStates(&rtM->solverInfo);
    ++rtM->Timing.clockTick0;
    rtM->Timing.t[0] = rtsiGetSolverStopTime(&rtM->solverInfo);
  }
}
```

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

```
void State_derivatives(void)
{
  XDot *_rtXdot;
  _rtXdot = ((XDot *) rtM->derivs);
  _rtXdot->ReplicaOfSource_CSTATE[0] = 0.0;
  _rtXdot->ReplicaOfSource_CSTATE[1] = 0.0;
  _rtXdot->ReplicaOfSource_CSTATE[0] += rtX.ReplicaOfSource_CSTATE[1];
  _rtXdot->ReplicaOfSource_CSTATE[0] += 0.4 * rtDW.DTC_input_1;
  _rtXdot->ReplicaOfSource_CSTATE[1] += rtDW.DTC_input_1;
}
void State_initialize(void)
{
  {
    rtsiSetSimTimeStepPtr(&rtM->solverInfo, &rtM->Timing.simTimeStep);
    rtsiSetTPtr(&rtM->solverInfo, &rtmGetTPtr(rtM));
    rtsiSetStepSizePtr(&rtM->solverInfo, &rtM->Timing.stepSize0);
    rtsiSetdXPtr(&rtM->solverInfo, &rtM->derivs);
    rtsiSetContStatesPtr(&rtM->solverInfo, (real_T **) &rtM->contStates);
    rtsiSetNumContStatesPtr(&rtM->solverInfo, &rtM->Sizes.numContStates);
    rtsiSetNumPeriodicContStatesPtr(&rtM->solverInfo,
      &rtM->Sizes.numPeriodicContStates);
    rtsiSetPeriodicContStateIndicesPtr(&rtM->solverInfo,
      &rtM->periodicContStateIndices);
    rtsiSetPeriodicContStateRangesPtr(&rtM->solverInfo,
      &rtM->periodicContStateRanges);
    rtsiSetErrorStatusPtr(&rtM->solverInfo, (&rtmGetErrorStatus(rtM)));
    rtsiSetRTModelPtr(&rtM->solverInfo, rtM);
  }

  rtsiSetSimTimeStep(&rtM->solverInfo, MAJOR_TIME_STEP);
  rtM->intgData.y = rtM->odeY;
  rtM->intgData.f[0] = rtM->odeF[0];
  rtM->intgData.f[1] = rtM->odeF[1];
  rtM->intgData.f[2] = rtM->odeF[2];
  rtM->contStates = ((X *) &rtX);
  rtsiSetSolverData(&rtM->solverInfo, (void *)&rtM->intgData);
  rtsiSetSolverName(&rtM->solverInfo,"ode3");
  rtmSetTPtr(rtM, &rtM->Timing.tArray[0]);
  rtM->Timing.stepSize0 = 2.0;
  rtX.ReplicaOfSource_CSTATE[0] = 0.0;
  rtX.ReplicaOfSource_CSTATE[1] = 0.0;
}
```

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

Problem 3. (10 points) Work out Problem 1 in the Exercises of Chapter 9 in [LS15].
[LS15] Edward A. Lee and Sanjit A. Seshia, Introduction to Embedded Systems, A Cyber-Physical Systems Approach, Second Edition, http://LeeSeshia.org, ISBN 978-1-312-42740-2, 2015.

Answer

Suppose the below program is executing on a 32-bit processor with a direct mapped cache with parameters (m, S, E, B) = (32, 8, 1, 8)

- **Consider the case when N = 16, N = 32, how many cache misses will be there?**
  - The block size is 2^8 bytes. One integer is 4 bytes long.So, 1 block here can store $(2^8/4) = 64$ integers.
  - The array is accessed 3 times in the program.
  - When N = 16, then total cache miss will be only 1 i.e. the first time the element is accessed, it will not be present in cache and will be fetched from the main memory and while fetching cache will fetch the entire block data in the cache and hence, all the 16 elements of the cache will be present here.
  - When N = 32, then also number of cache misses will be **1** as the block can accommodate 64 integers. The cache miss will happen while accessing the array for the first time, post which the entire array will be fetched inside the cache. Also, when the program access the array for the 2nd and 3rd time, that data will be available in the cache hence no cache misses later on. Total misses = 1
- **c) Now executing for N = 16 on a 2-way set-associative cache with parameters (m, S, E, B) = (32, 8, 2, 4). In other words, the block size is halved, while there are two cache lines per set. How many cache misses the code suffers?**
  - In this scenario, the block size will be 2^4 bytes. So, 1 block can store $2^4/4 = 4$ numbers
  - Tag bits (bits that uniquely determine block that is stored in the cache line) = m - s - b = 32-8-4 = 20 bits. Hence, in a cache line, 2^20 blocks are possible, each with the capacity to store 4 numbers.
  - There will be total of 4 cache misses during access of 1st, 5th, 9th and 13th elements
  - When ever there is a cache miss, the entire block of data is fetched from the main memory, hence after 1st cache miss, 2nd, 3rd, 4th will be available in the cache. Then, 5th one will be a miss and cache will fetch entire block (currently block can hold 4 numbers) from the main memory and so on.
  - As we have seen in the C code, that the array is accessed thrice, We can say that when ever the array is being accessed from the cache for the second time, the data of the entire array is already available, hence there will be no cache misses later on. **Total misses = 4**

Name: Ajita Shree

Roll No e.g. 170001: 20111262

Dept. e.g. CSE: Computer Science Engineering

Indian Institute of Technology Kanpur
CS637 Embedded and Cyber-Physical Systems
Homework Assignment 3
Deadline: October 25, 2020

BLANK SPACE: Any answers written here will be left ungraded.
No exceptions.
You may use this space for rough work.