K means accelerator

## Introduction

### The K means algorithm

The K means algorithm is an iterative algorithm which divides a given data vector to K different clusters (K is a natural number). Each cluster will be characterized by its “center of mass”, what will be referred in this paper as centroid.

#### The algorithm steps

For a simpler explanation, we can assume K is a constant predefined natural value. First, will define some symbols:

-the cluster number "*i*" centroid

– the group of points in cluster number "*i*"

##### Initiation step

The first step in the algorithm is to randomly choose centroids for the K clusters. The “time” (“*t*”) for the initialization step will be defined as zero.

##### Classification step

In each iteration(time) of the algorithm we first assign each points from the input vector to a cluster based on the “distance” from the point to the clusters centroid. A point will be assigned to cluster number “i” if the metrical distance between it and the cluster’s centroid is the minimum between the distances from the point to all others cluster’s centroids. To simplify:

\*In case of the distance from two different clusters is the same and is the minimum found, the chosen cluster is the one with the lowest index.

##### Centroids update step

After the classification step, the centroids of each cluster are updated to be mean of all points which belong to it in end of iteration(time) *t*. This is done by verifying if a cluster is empty(in this case the centroid is not changed) and then calculating the mean of all the clusters points:

##### Converge check step

If the centroids of the next iteration calculated in the step above are all equal to the current centroids, then the algorithm comes to an end. Else, the iteration number(time) is increased by one and a new iteration begins with the assigning step.

Note : the k means algorithm assures convergence to a local minimum.

#### Choosing K

Usually the optimal K is not known before the beginning of the algorithm. Therefore, a an error parameter can be defined to help choosing K. The most commonly known error parameter is the clustering error which is defined by:

In this formula, the elements are:

As K increases, the error decreases. For example, if K is as the number of pints in the input vector, the error will be zero. This because it cluster will have just one point which will also be its centroid, but in this case no new information was added by the algorithm.

One suggested method of choosing a natural K so the clustering error is minimized is by gradually increasing K and calculating for each increasement. The process ends when the error reaches a value so that , where is a predefined threshold.

### AMBA APB

#### Introduction

The Advanced Peripheral Bus (APB) is part of the Advanced Microprocessor Bus Architecture (AMBA) protocol family. This protocol is a single master multi slave and set guidelines for transactions between the master and its low-bandwidth peripherals, the slaves. The APB protocol signal transactions are only related to the rising edge of the clock and every transaction takes at least two cycles. It can be used to provide access to the programmable control registers of peripheral devices. Furthermore, the APB is a low-cost interface that is optimal for minimal power consumption.

The figure bellow (Key to timing diagram conventions) explains the timing diagrams in the following sections. Shaded bus and signal areas are undefined, so the bus or signal can assume any value within the shaded area at that time. The actual level is unimportant and does not affect normal operation.



Figure : Key to timing diagram of APB protocol

The signals which are part of APB protocol are listed and described in the table below:

|  |  |  |
| --- | --- | --- |
| Signal | Source | Description |
| PCLK | Clock source | Clock. The rising edge of PCLK times all transfers on the APB. |
| PRESETn | System bus equivalent | Reset. The APB reset signal is active LOW. This signal is normally connected  directly to the system bus reset signal. |
| PADDR | Master | Address. This is the APB address bus. It can be up to 32 bits wide and is driven  by the peripheral bus bridge unit. |
| PSELx | Master | Select. The APB bridge unit generates this signal to each peripheral bus slave.  It indicates that the slave device is selected and that a data transfer is required.  There is a PSELx signal for each slave. |
| PENABLE | Master | Enable. This signal indicates the second and subsequent cycles of an APB  transfer. |
| PWRITE | Master | Direction. This signal indicates an APB write access when HIGH and an APB  read access when LOW. |
| PWDATA | Master | Write data. This bus is driven by the peripheral bus bridge unit during write  cycles when PWRITE is HIGH. This bus can be up to 32 bits wide. |
| PREADY | Slave | Ready. The slave uses this signal to extend an APB transfer. |
| PRDATA | Slave | Read Data. The selected slave drives this bus during read cycles when  PWRITE is LOW. This bus can be up to 32-bits wide. |
| PSLVERR | Slave | This signal indicates a transfer failure. APB peripherals are not required to  support the PSLVERR pin. This is true for both existing and new APB  peripheral designs. Where a peripheral does not include this pin then the  appropriate input to the APB bridge is tied LOW. |

Table : APB signal description

The PADDR,PWRITE,PWDATA signals are common among all the slaves, however there are as many PSEL signals as slaves, and for each slave one PRDATA from it to the master. The following shows the block diagram between master and slave of APB:



Figure : APB block diagram

#### Operating states

The figure bellow describes the operating states of the protocol:



Figure : APB operating states

The state machine operates through the following states:

**IDLE** - This is the default state of the APB.

**SETUP** - When a transfer is required the bus moves into the SETUP state, where the appropriate select signal, PSELx, is asserted. The bus only remains in the SETUP state for one clock cycle and always moves to the ACCESS state on the next rising edge of the clock.

**ACCESS** - The enable signal, PENABLE, is asserted in the ACCESS state. The

address, write, select, and write data signals must remain stable during

the transition from the SETUP to ACCESS state. Exit from the ACCESS state is controlled by the PREADY signal from the slave:

• If PREADY is held LOW by the slave then the peripheral bus remains in the ACCESS state.

• If PREADY is driven HIGH by the slave then the ACCESS state is exited and the bus returns to the IDLE state if no more transfers are required. Alternatively, the bus moves directly to the SETUP state if another transfer follows.

#### Transfers

Each transfer consists of two cycles: one for the SETUP state and another for the ACCESS state. There are three types of transfers: write transfers, read transfers and error response transfers. In addition, write and read transfers can be with or without wait states, that are SETUP states which follow an ACCESS state instead of going to IDLE STATE.

##### Write Transfers

###### Write Transfers without wait states

A write transfer without wait states consist of two clock cycles: in the first (the SETUP STATE) the signals: address (PADDR), write data (PWDATA), write (PWRITE) and select (PSEL) are asserted. PADDR is asserted to the desired address where the data is supposed to be written, PWDATA is asserted to the desired data to be written, PWRITE is asserted HIGH and PSEL is asserted HIGH only for the specific slave which the write command is for, the rest of the PSEL lines are driven LOW. These signals remain unchanged through the second cycle.

In the second cycle (the ACCESS state) the slave sets the enable signal (PENABLE) HIGH. The ready signal (PREADY) is set HIGH by the slave in order the informed the master that the slave is ready to receive the data, which is latched by the slave in the rising edge ending the second clock cycle. After this last clock rising edge, PREADY is driven LOW by the slave, PENABLE is driven LOW by the master, and PSEL is driven LOW by the master (unless the transfer is to be followed immediately by another transfer to the same slave, in which case the signals PENABLE and PSEL remain as they are) meaning that the transfer is over.

In Figure 4 an example of write transaction with no wait states can be seen, with the first cycle of the transfer being from T1 to T2 and the second cycle from T2 to T3.



Figure : APB write transfer with no waits

###### Write Transfers with wait states

The first cycle of the transfers is the as the transfers without wait states. During the ACCESS state, when PENABLE is HIGH, the transfer can be extended by driving the PREADY LOW. The signals PADDR, PWRITE, PSEL, PENABLE and PDATA remain unchanged from the end of the first cycle (SETUP state) until the data is latched by the slave, which occurs at the first rising clock edge after the slave sets the PREADY signal HIGH. After this clock rising edge, PREADY is driven LOW by the slave, PENABLE is driven LOW by the master, and PSEL is driven LOW by the master meaning that the transfer is over (unless the transfer is to be followed immediately by another transfer to the same slave, in which case the signals PENABLE and PSEL remain as they are) .

In Figure 5 an example of write transaction with wait states can be seen, with the first cycle of the transfer being from T1 to T2,two wait states occur from T2 until T4 and the last cycle of the transfer from T4 to T5 , in which the slave sets the PREADY signal HIGH and at the end of this cycle the data is latched by the slave.



Figure : APB write transfer with wait states.

##### Read Transfers

###### Read Transfers without wait states

A read transfer without wait states consist of two clock cycles: in the first (the SETUP STATE) the signals: address (PADDR), write (PWRITE) and select (PSEL) are asserted. PADDR is asserted to the desired address where the data is supposed to be read, PWRITE is asserted LOW and PSEL is asserted HIGH only for the specific slave which the write command is for, the rest of the PSEL lines are driven LOW. These signals remain unchanged through the second cycle.

In the second cycle (the ACCESS state) the slave sets the enable signal (PENABLE) HIGH. The PRDATA signal is set by the slave according to the data in stored in the desired address(the address which is set in PADDR signal) and the ready signal (PREADY) is set HIGH by the slave in order the informed the master that the slave is ready to send the data. The data in PRDATA signal is latched by the master in the rising edge ending the second clock cycle. After this last clock rising edge, PREADY is driven LOW by the slave, PENABLE is driven LOW by the master, and PSEL is driven LOW by the master (unless the transfer is to be followed immediately by another transfer to the same slave, in which case the signals PENABLE and PSEL remain as they are) meaning that the transfer is over.

In Figure 6 an example of write transaction with no wait states can be seen, with the first cycle of the transfer being from T1 to T2 and the second cycle from T2 to T3.



Figure 6: APB read transfers with no wait states

###### Read Transfers with wait states

The first cycle of the transfers is the as the first cycle of transfer without wait states. During the ACCESS state, when PENABLE is HIGH, the transfer can be extended by driving the PREADY LOW. The signals PADDR, PWRITE, PSEL and PENABLE remain unchanged from the end of the first cycle (SETUP state) until the data is latched by the master, which occurs at the first rising clock edge after the slave sets the PREADY signal HIGH. After this clock rising edge, PREADY is driven LOW by the slave, PENABLE is driven LOW by the master, and PSEL is driven LOW by the master meaning that the transfer is over (unless the transfer is to be followed immediately by another transfer to the same slave, in which case the signals PENABLE and PSEL remain as they are) .

In Figure 7 an example of read transaction with wait states can be seen, with the first cycle of the transfer being from T1 to T2,two wait states occur from T2 until T4 and the last cycle of the transfer from T4 to T5 , in which the slave sets the PREADY signal HIGH and at the end of this cycle the data is latched by the master.



Figure :APB read transfer with wait states.

##### Error response

Some APB peripheral offer a way of indicating that an error occurred during a transfer with the PSLVERR signal. Errors can occur both in read and write transfers, and the signal PSLVERR is only considered valid during the last cycle of an APB transfer, when PSEL, PENABLE, and PREADY are all HIGH.

It is recommended, but not mandatory, that you drive PSLVERR LOW when it is not

being sampled. That is, when any of PSEL, PENABLE, or PREADY are LOW.

Transactions that receive an error, might or might not have changed the state of the

slave. This is peripheral-specific, and either is acceptable.

When a write transaction receives an error, this does not mean that the register within the slave has not been updated. Read transactions that receive an error can return invalid data.

There is no requirement for the slave to drive the data bus to all 0s for a read error.

APB slaves are not required to support the PSLVERR pin. This is true for both

existing and new APB peripheral designs. Where a slave does not include this pin

then the appropriate input to the master is tied LOW.

###### Error response in a write transfer

When there is an error in a write transfer and the slave in the transfer has an active PSLVERR signal, during the last cycle of the transfer (when PSEL, PENABLE, and PREADY are all HIGH) PSLVERR is driven HIGH, informing the master about the error in the transaction. These can be seen in Figure 8:



Figure : APB error in write transfer

###### Error response in a write transfer

When there is an error in a read transfer and the slave in the transfer has an active PSLVERR signal, during the last cycle of the transfer (when PSEL, PENABLE, and PREADY are all HIGH) PSLVERR is driven HIGH, informing the master about the error in the transaction. These can be seen in Figure 9:



Figure : APB read in write transfer

## Bibliography

* K means algorithm – tutorial 3 of Technion winter 2018/2019 machine learning course 046195
* AMBA APB – ARM AMBA 3 APB Protocol v1.0 Specification
* AMBA APB – International Journal of Scientific Engineering and Applied Science (IJSEAS) - Volume-1, Issue-3, June 2015 ISSN: 2395-3470 www.ijseas.com 492

Design & Implementation of Advance Peripheral Bus Protocol