SGS-THOMSON MICROELECTRONICS

APPLICATION NOTE

CASCADING IMSA110s

1. INTRODUCTION

The IMSA110 is a single-chip programmable and cascadable device suitable for many high speed image and signal processing applications. It consists of a configurable array of multiply-accumulators (420 MOPs), three programmable length 1120 tage shift registers, a versatile post-processing unit and a microprocessor interface for configuration and control purposes. The comprehensive on-chip facilities makes a single device capable of dealing with many image processing operations. A simplified block diagram is shown in Figure 1.

For some applications however, the power and versatility of a single IMSA110 is not sufficient, in these cases a cascade of devices often provides a solution. The purpose of this document is to describe some of the most useful ways to cascade IMSA110s to achieve even higher performance and as such does not cover the use of the backend processor or device applications.

2. OPERATION OF A SINGLE IMSA110

The A110 may be set up as either a one or two dimensional multiplier accumulator array (MAC).

2.1 One dimensional operation of an IMSA110

For one dimensional operation the first delay PSRc is set to some arbitrary value (normally zero) while PSRb and PSRa are set to zero. N.B. at any given point in time the first MAC stage in bank c is

processing the oldest data while the last MAC stage of bank a is processing the newest data.

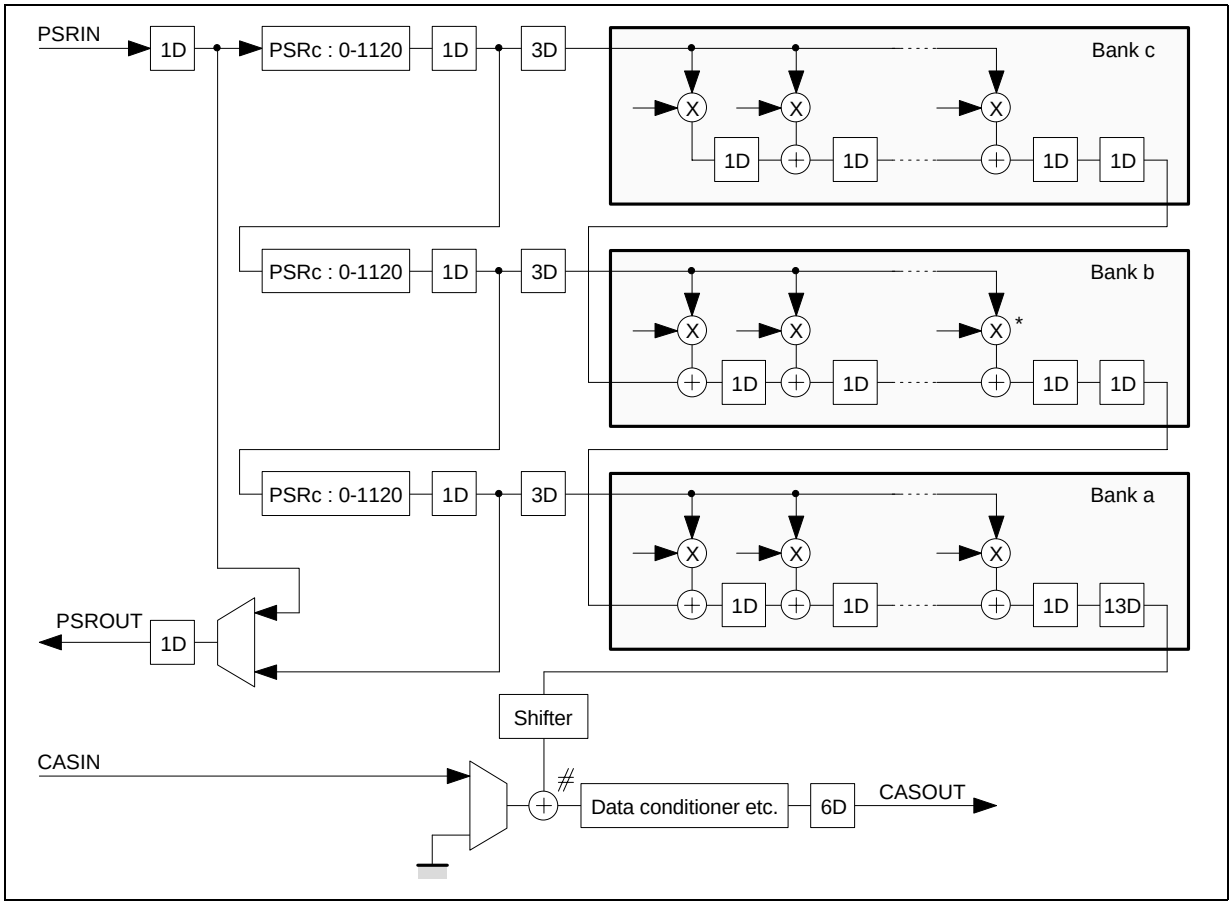
2.2 Two dimensional operation of an IMSA110

For two dimensional operation the first delay (PSRc) is again set to some arbitrary value; however, the setting of PSRa and PSRb is dependant on the line length in pixels of the image being processed. It turns out that in order to achieve a rectangular convolution window the number of delays to be programmed into PSRa and PSRb is equal to the line length in pixels plus the length of the MAC pipelines (seven stages). For example if the screen width of the image to be processed is 512 pixels then the delay to be programmed into shift registers PSRa and PSRb is 519.

N.B. normally when processing an image with an arbitrary setting of PSRc the delay (latency) through the IMSA110 causes the output image to be incorrectly aligned or skewed. This results in an apparent rotation of the output image in the horizontal plane. To correct this problem PSRc may be adjusted to introduce a suitable number of delays to shift the image into the correct position.

Typically image data is fed into an IMSA110 line by line starting at the top left and ending at the bottom right. Given this definition it may be seen that the first MAC stage in each row is processing the data nearest the left hand side of the screen (the oldest data) and that the last MAC stage in each row is processing the data nearest the right hand side of the screen (the newest data). In a similar fashion the first row is always processing the newest data (the data nearest the bottom of the screen) and the last row is always processing the oldest data (the data nearest the top of the screen). It is important to bear in mind these relationships when programming IMSA110s, otherwise the operation being performed on an image may not be what was expected.

Figure 1 : Block Diagram of the IMSA110s



3. FUNDAMENTALS OF CASCADING IMSA110s

Consider a single IMSA110 configured to perform some task on a stream of data values. The filter kernel formed by the coefficients may be thought of as a block passing over the data. To produce bigger filters it is necessary to join a number of separate blocks together. This may be achieved by connecting together a number of IMSA110s, as shown in Figure 2, and configuring them suitably. In order to create a contiguous filter kernel (i.e. a filter without overlap or gaps) it is essential that the route between PSRin and PSRout for each device is programmed correctly and that the internal delay lines are programmed to the correct lengths.

To assist in the calculation of the delays to be programmed into the programmable shift registers it is convenient to define a reference data path through the MAC of any given IMSA110. In this document, unless specified, the reference path is taken to be from the input to the multiplier marked with an asterisk (*) in Figure 1 to the cascade adder marked with a hash (#).

In addition before embarking on any calculation it is necessary to know the following :

1 The delay between PSRin and PSRout when the data is routed directly from PSRin to PSRout without passing through the programmable shift registers. This delay is known as D_D.
2 The delay along the reference path. This delay is known as D_R.
3 The delay through the backend between cascade in and cascade out. This delay is known as D_B.
4 The locations of the other inherent delays within IMSA110s.
5 The meaning of line length, kernel width and kernel height. See Figure 3 for a definition of these terms.

Figure 1 shows a functional block diagram of an IMSA110 with all the inherent delays included. From this diagram it is possible to calculate the value of the three delay constants as shown in table 1.

Table 1

D_D=1+1	D_R=(1+1)+(7+1)+(7+13)	D_B=6
D_D=2	D_R=30	

Figure 2 : Standard Connection for Cascading IMSA110s

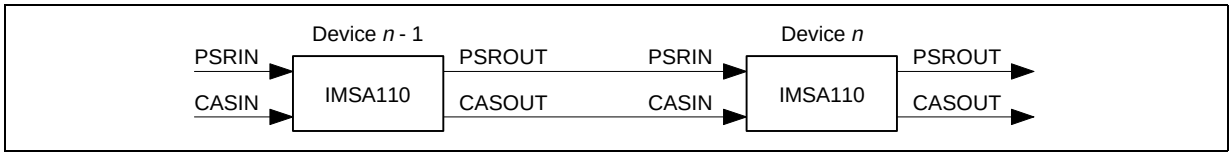
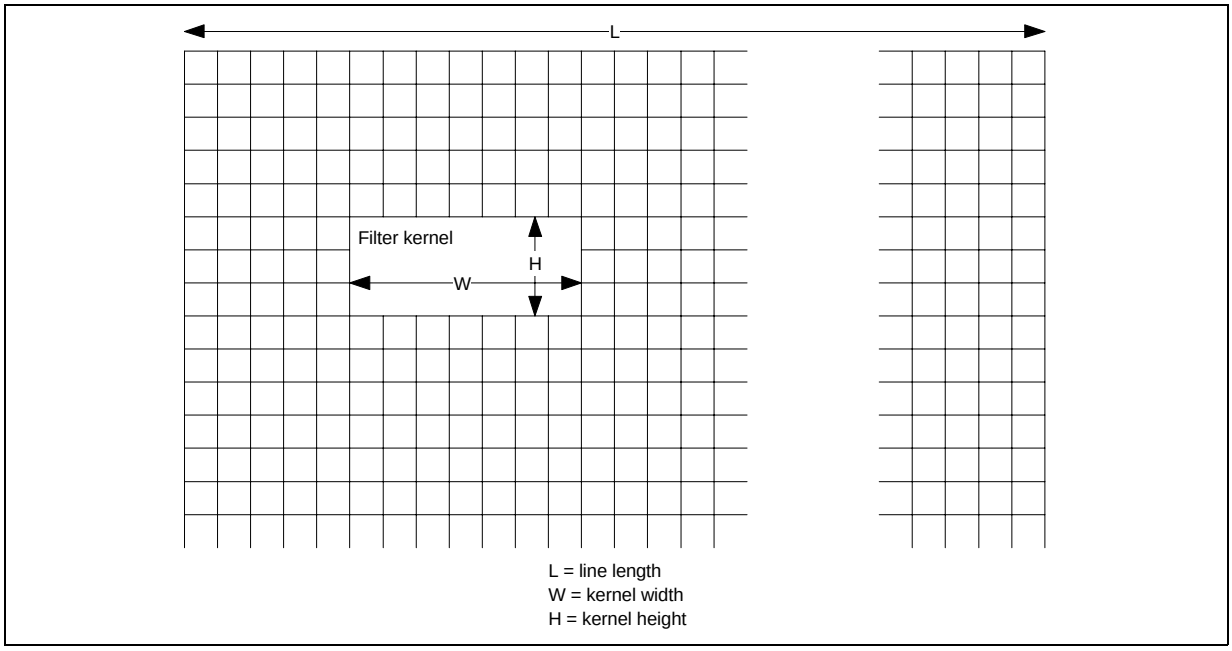


Figure 3 : Depiction of Line Length, Kernel Width and Kernel Height



4. CASCADING IMSA110s TO PRODUCE LONG ONE DIMENSIONAL FILTERS

A single IMSA110 is capable of producing a one dimensional filter with up to 21 taps (shorter filters may be made by setting unrequired coefficients to zero). To create longer filters it is necessary to cascade a number of IMSA110s together. Each additional device added to the cascade gives an additional 21 taps allowing filters of almost unlimited size to be built from simple building blocks.

To develop the delays required to be set up in a one dimensional cascade the system shown in Figure 4 will be considered. This system only contains two devices but will be examined in a general way so that rules may be developed for cascades of arbitrary length. It has already been mentioned how to set up the delays to achieve one dimensional convolution in a single device. Fortunately, in cascades of IMSA110s the data relationships within each device are the same as those which would exist inside a single non cascaded device processing the same data. Hence, in the one dimensional cascade under consideration the delays programmed into PSRa and PSRb of each device are zero.

In order to cascade IMSA110s into long one dimensional filters the data is normally routed directly from the input to the output of each device without passing through the programmable shift registers, as shown in Figure 4. It may be seen that each piece of data takes two routes through the cascade. One route generates partial results via the MAC of device n-1 and the other via the MAC of device n. These partial results are eventually combined at the cascade adder in the backend of device n. To produce the correct result it is important that these two separate data streams are aligned correctly.

Assuming that the delay in the PSRc of device n-1 is x_{n-1} and that the delay in PSRc of device n is x_n, it is desired to calculate the relationship between these delays for correct combination of the partial results. Consider an item of data when it reaches device n-1. The delay before the component due to this data, flowing via the reference path in device n-1, reaches the cascade adder of device n is:

$$D_{n-1} = 1+x_{n-1}+1+3+D_R+D_B$$
$$D_{n-1} = 41+x_{n-1}$$

Similarly the delay before the component due to this data, flowing via the reference path in device n, reaches the cascade adder of device n is:

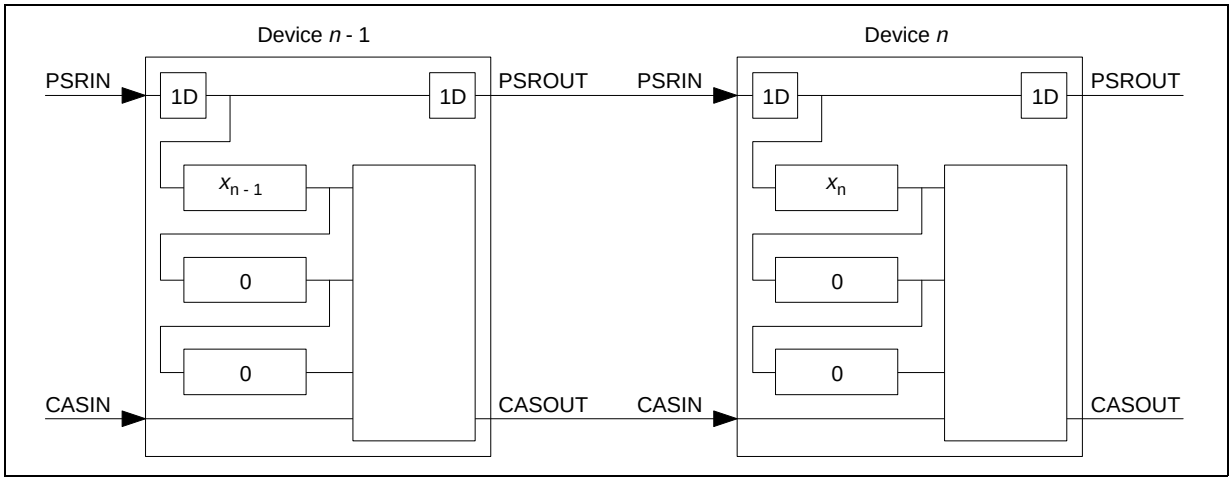
$$D_n = D_D+1+x_n+1+3+D_R$$
$$D_n = 37+x_n$$

Now, for a contiguous convolution kernel, it is desired for the results flowing via the MAC of device n-1 to arrive at the cascade adder of device n, 21 clock cycles behind those which have come from the other route. Hence:

$$D_{n-1}-D_n = 21$$
$$41+x_{n-1}-37-x_n = 21$$
$$x_{n-1} = x_n+17$$

This means that the PSRc of device n-1 must be programmed with the value which is in PSRc of device n plus a fixed constant of 17. This rule may be extended to take into account any number of devices providing that the maximum length of the delay lines is not exceeded. The PSRc of the last device in the cascade may be programmed to an arbitrary value (normally zero) providing the maximum length of the first PSRc delay in the cascade is not exceeded.

Figure 4 : Direct Data Path Connection for Cascading IMSA110s



For example consider the problem of filtering a data stream with a 50 tap filter. This could be achieved by cascading three IMSA110s. Typical delays which would have to be programmed into the devices are given in table 2.

Table 2

	Device 1	Device 1	Device 2
PSRa	0	0	0
PSRb	0	0	0
PSRc	34	17	0

5. CASCADING IMSA110s TO PRODUCE WIDER TWO DIMENSIONAL FILTERS

A single IMSA110 is capable of filtering an image with a two dimensional kernel which has a maximum width of seven cells (narrower filters may be made by setting unrequired coefficients to zero). To create wider filters it is necessary to cascade a number of IMSA110s together. Each additional device added to the cascade increases the maximum width by an additional 7 cells, allowing filters of almost unlimited width to be created.

The connections required to cascade IMSA110s into horizontal cascades may be seen in Figure 4. It may be noted that the connections for this type of cascade are identical to those presented in section 4 for one dimensional cascading. The difference in function is achieved by changing the delays present in the programmable shift registers. It was mentioned in section 2 that for two dimensional filtering using a single device the length of PSRa and PSRb have to be programmed to the line length plus seven. Hence to ensure correct alignment of the rows of the filter in a horizontal cascade it is necessary that PSRa and PSRb of each of the devices must also be set to this value. In order to cascade horizontally the pixel data is normally routed directly from the input to the output of each device without passing through the programmable shift registers. As before it may be seen that each item of data (pixel) takes two routes through the cascade. By assuming that the delay in the PSRc of device n-1 is x_{n-1} and that the delay in PSRc of device n is x_n, then the route delay equations derived are the same as those calculated in section 4.

$$D_{n-1} = 41+x_{n-1}$$
$$D_n = 37+x_n$$

Now, for a contiguous convolution kernel, it is desired for results flowing via the MAC of device n-1 to arrive, at the cascade adder of device n, 7 clock cycles behind those which have come from the other route. This may be achieved by ensuring that the data passing via MAC n-1 takes 7 cycles longer than data passing via the MAC n route. Hence:

$$D_{n-1}-D_n = 7$$
$$41+x_{n-1}-37-x_n = 7$$
$$x_{n-1} = x_n+3$$

This means that the PSRc of device n-1 must be programmed with the value which is in PSRc of device n plus a fixed constant of 3. This rule may be extended to cascade any number of devices providing that the maximum length of the delay lines is not exceeded. The value programmed into the PSRc of the last device in the cascade is arbitrary (normally adjusted to deskew the output image) but must not be set so high that the PSRc of the first device in the cascade exceeds its maximum.

For example consider the problem of filtering a 1024 pixel wide image with a 15×3 filter kernel. This could be achieved by cascading three IMSA110s into a horizontal cascade. Typical delays which would have to be programmed into the devices are given in table 3.

Table 3

	Device 1	Device 2	Device 3
PSRa	1031	1031	1031
PSRb	1031	1031	1031
PSRc	6	3	0

6. CASCADING IMSA110s TO PRODUCE HIGHER TWO DIMENSIONAL FILTERS

The maximum height of a two dimensional filter kernel produced by a single IMSA110 is three cells. This is restricting in some applications but, may be easily overcome by cascading a number of IMSA110s into a single vertical strip. The theoretical maximum height of filter which can be created is equal to three times the number of devices cascaded. Hence the vertical filter size is limited only by the number of devices used.

To develop the delays required to be setup in a vertical cascade the system shown in Figure 5 will be considered. This system only contains two devices but will be examined in a general way so that rules may be developed for cascades of arbitrary length. It was mentioned in section 2 that for two dimensional filtering using a single device the length of PSRa and PSRb have to be programmed to the line length plus seven (L+7). Obviously to ensure correct alignment of the rows of the filter in a vertical cascade it is necessary that PSRa and PSRb of each of the devices must also be set to this value.

To cascade vertically the pixel data is normally routed from the input to the output of each device via the programmable shift registers (see Figure 5). Again it may be seen that each pixel takes two routes through the cascade. One route generates partial results via the MAC of device n-1 and the other via the MAC of device n.

These partial results are eventually combined at the cascade adder in the backend of device n. In order to produce the correct result it is important that these two data streams are aligned correctly.

Assuming that the delay in the PSRc of device n-1 is x_{n-1} and that the delay in PSRc of device n is x_n, it is desired to calculate the the relationship between these delays for correct combination of the partial results. Consider a pixel when it reaches device n-1. The delay before the component due to this pixel, flowing via the reference path in device n-1, reaches the cascade adder of device n is:

$$D_{n-1} = 1+x_{n-1}+1+3+D_R+D_B$$
$$D_{n-1} = 41+x_{n-1}$$

Similarly the delay before the component due to this pixel, flowing via the reference path in device n, reaches the cascade adder of device n is:

$$D_n = 1+(x_{n-1}+1)+(L+7+1)+(L+7+1)+1+1+(x_n+1)+3+D_R$$
$$D_n = 54+2L+x_{n-1}+x_n$$

But for a contiguous convolution kernel it is desired for results flowing via MAC n to arrive, at the cascade adder of device n, three line lengths after those which have come from the other route. This may be achieved by ensuring that the data passing via MAC n takes 3L (where L is the line length in pixels) cycles longer than data passing via the MAC n-1 route. Hence:

$$D_n-D_{n-1}=3L$$
$$54+2L+x_{n-1}+x_n-41-x_{n-1}=3L$$
$$x_n=L-13$$

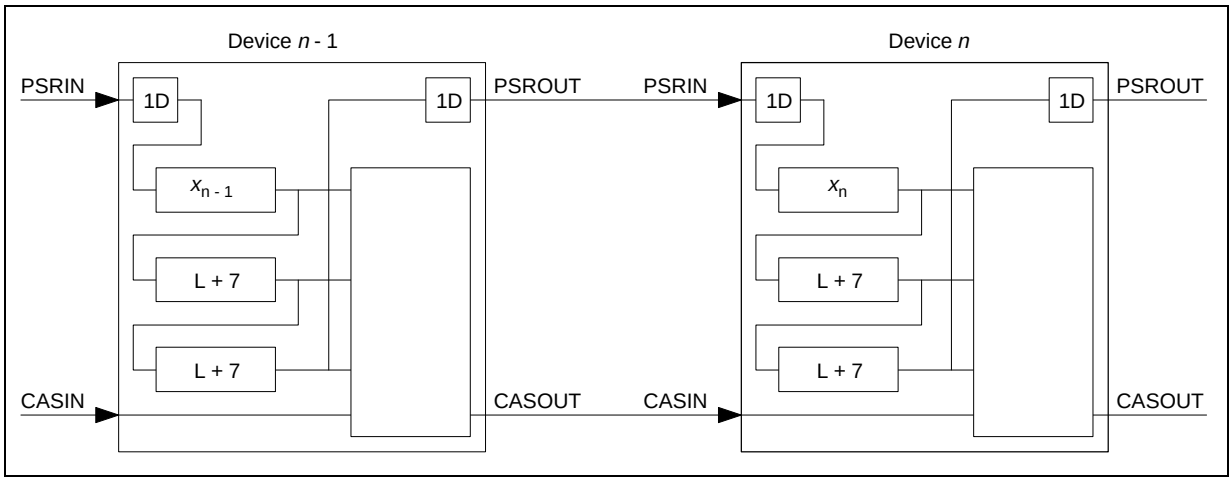
This means that the PSRc of device n must be programmed with a value which is equal to the line length minus a fixed constant of 13. This rule may be extended to cascades containing any number of devices providing that the maximum length of the delay lines is not exceeded. N.B. the setting of the PSRc of the first device in the cascade is arbitrary and may be adjusted to deskew the output image.

For example consider the problem of filtering a 512 pixel wide image with a 7×7 filter kernel. This could be achieved by cascading 3 IMS A110s into a vertical cascade. Typical delays which would have to be programmed into the devices are given in table 4.

Table 4

	Device 1	Device 2	Device 3
PSRa	519	519	519
PSRb	519	519	519
PSRc	0	499	499

Figure 5 : Indirect Data Path Connection for Cascading IMSA110s



7. Cascading IMSA110s to produce wider and higher two dimensional filters

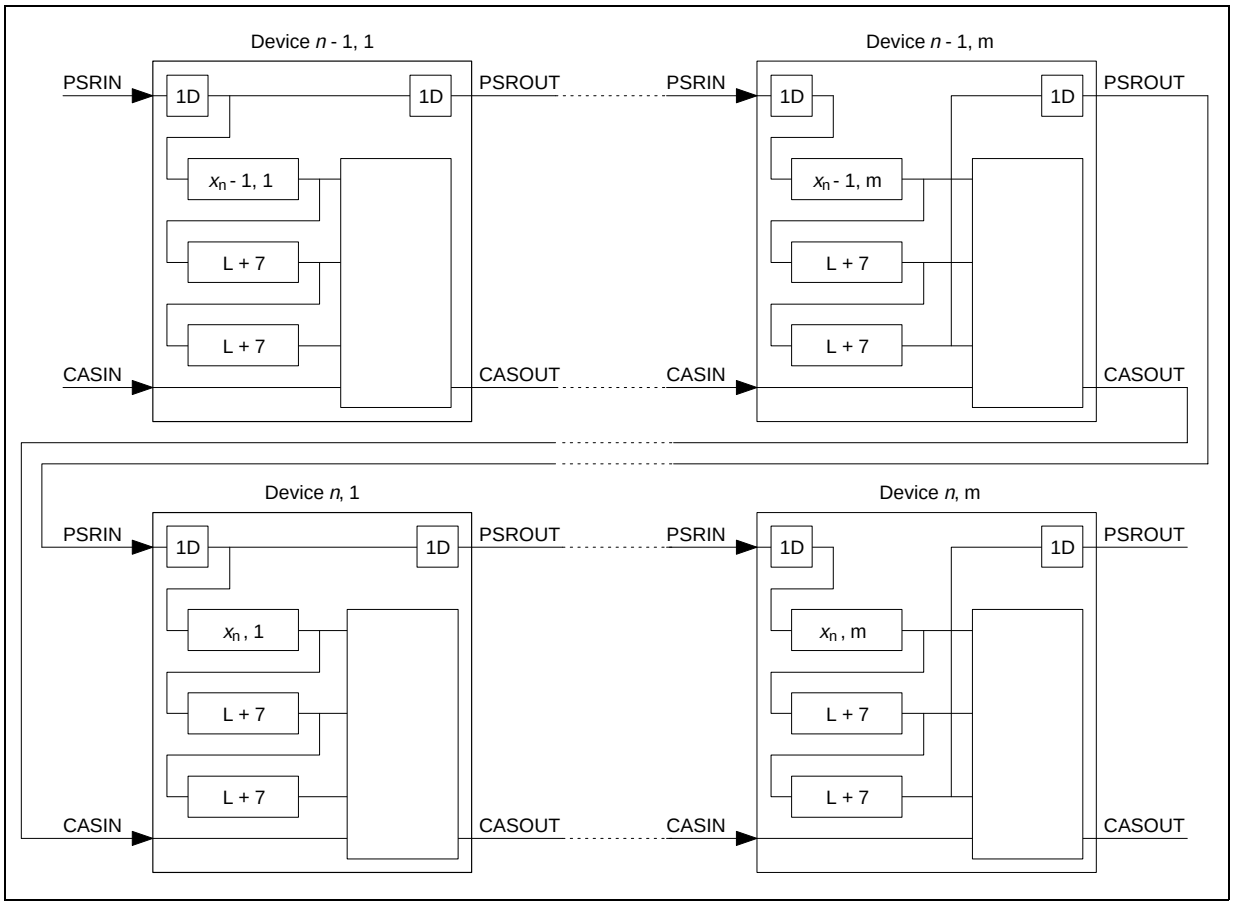
To produce filters which are both wider and higher than allowed by a single IMSA110 it is possible to cascade a number of the wider filters discussed in section 5 into a vertical strip.

The connections required to cascade IMSA110s into two dimensional cascades may be seen in Figure 5. The system shown has arbitrary width but only two rows of devices allowing a maximum filter height of six cells. However the system will be examined in a general way so that rules may be developed for cascades of arbitrary height. It may be noted that across each row, except for the last device, direct connection is used between PSRin and PSRout. The last device uses the indirect route via the programmable shift registers to connect to the first device of the next row. Since each row of this cascade consists of a horizontal cascade the rules developed for the delays in such a cascade (see section 6) apply to each row of this larger configuration. However, the relationship between the delays in the vertical direction requires careful consideration.

Assuming that the array of IMSA110s contains M devices in the horizontal direction and that the delay in PSRc of each device is as shown in Figure 6, it is desired to calculate the relationship between these delays for correct combination of the partial results generated by each row within the cascade. Consider a pixel when it reaches device n-1,1. The delay before the first component due to this pixel, flowing via the reference path in device n-1,1, reaches the cascade adder of device n,1 is:

$$D_{n-1,1} = 1+x_{n-1,1}+1+3+D_R+D_BM$$
$$D_{n-1,1} = 35+6M+x_{n-1,1}$$

Figure 6 : Connections for Cascading IMSA110s into Wider and Higher 2-D Filters



Similarly the delay before the component due to this pixel, flowing via the reference path in device n,1, reaches the cascade adder of device n,1 is:

$$D_{n,1} = 2\,(M-1) + 1 + (x_{n-1}\,,\, M+1) + (L+7+1) + (L+7+1) + 1 + 1 + (x_{n,1}+1) + 3 + D_R$$

$$D_{n,1} = 52+2M+2L+x_{n-1,M}+x_{n,1}$$

But for a contiguous convolution kernel it is desired for results flowing via MAC n,1 to arrive, at the cascade adder of device n,1 a period of 3L clock cycles after those which have come from the other route. Hence:

$$D_{n,1}-D_{n-1,1} = 3L$$

$$52+2M+2L+x_{n-1,M}+x_{n,1}-35-6M-x_{n-1,1} = 3L$$

$$17-L-4M+x_{n-1,M}+x_{n,1} = x_{n-1,1}$$

Now it is also known from section 5 that any given device in a row except the final device has PSRc programmed to 3 more than the device which follows. This leads to the following relationship between the delays programmed into the first and the last devices of the top row:

$$x_{n-1,1} = x_{n-1,M}+3(M-1)$$

By substituting this result into the previous result gives: $x_{n,1}=7M+L-20$

This means that the PSRc of device n,1 must be programmed with the value which is equal to 7 times the number of devices cascaded horizontally plus the line length minus a fixed constant of 20. This rule may be extended to cascades containing any number of devices providing that the maximum length of the delay lines is not exceeded. N.B. the setting of PSRc of the right most device in the first row is arbitrary, but is normally adjusted to deskew the output image.

For example consider the problem of filtering a 512 pixel wide image with a 9×9 filter kernel. This could be achieved by cascading six IMSA110s into a cascade containing three rows of two devices. Typical delays which would have to be programmed into the devices are given in Table 5.

8. Cascading IMSA110s to perform multi pass filtering operations

In addition to being able to cascade IMSA110s for increased filter size it is also possible to cascade devices to perform multi pass filtering operations.

For example consider the problem of edge detection in a noisy image. This task is often performed in two stages the first is low pass filtering to reduce the amount of noise and the second is the edge detection operation. This complete task may be performed by cascading two IMSA110s as shown in Figure 7. Note that only an eight bit window of CASout from the first device is connected to PSRin of the second device.

To configure such a cascade to perform the double filtering operation each device is considered separately and the delays are setup as described in section 2. For the example under consideration the coefficients of the first device are configured to perform the low pass filter operation while the coefficients of the second device are configured as an edge detector.

This technique of multi pass filtering can obviously be extended to include more devices or it may be combined with the cascading techniques discussed in earlier sections to allow multi pass filtering with larger filter sizes.

It is possible to use a single device for multi pass filtering. This technique works by feeding back alternate cascade outputs to PSRin, and making use of bank swapping. Figure 8 shows the basic setup. The disadvantages of this method are:

1 The maximum data throughput is halved.
2 The maximum filter size is reduced.
3 External logic is required.

To setup such a system requires careful programming to achieve the desired result. For example consider the problem of passing the local averaging filter kernel shown below over an image twice.

1	1	1
1	1	1
1	1	1

It may be shown using similar techniques to those presented earlier that the delays to be programmed into the programmable shift registers a and b are: 2L+7

This value is equal to twice the line length plus the length of the MAC pipelines. N.B. logical reasoning would have lead to the same result by considering that the data rate within the device is equal to twice the rate of the applied image data.

Table 5

	Device 1,1	Device 1,2	Device 2,1	Device 2,2	Device 3,1	Device 3,2
PSRa	519	519	519	519	519	519
PSRb	519	519	519	519	519	519
PSRc	3	0	506	503	506	503

Figure 7 : Cascading IMSA110s for Multi-pass Filtering

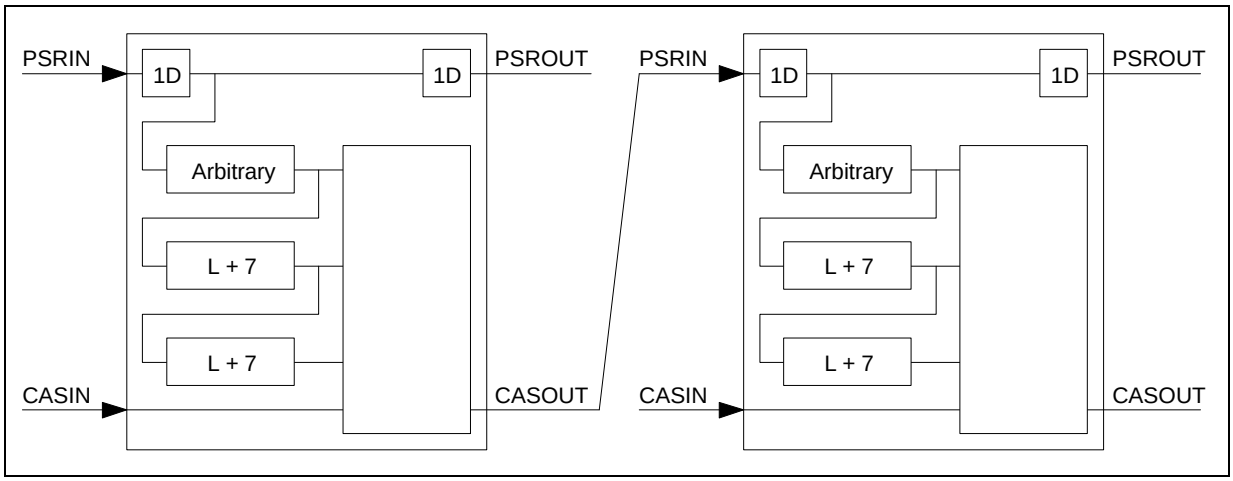
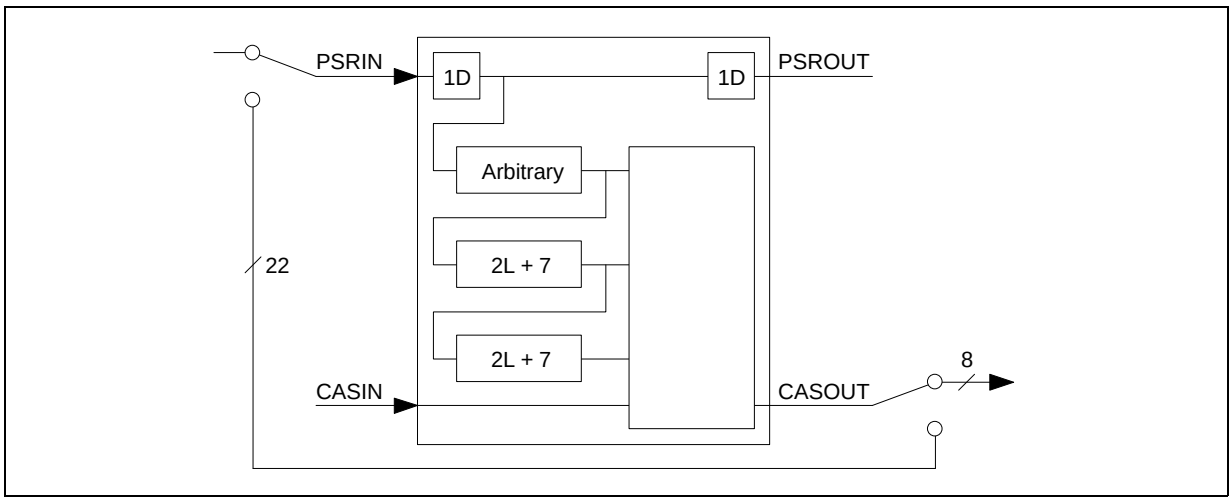


Figure 8 : Multi-pass Filtering by using Feedback



To create the correct filter kernels it is very important that the coefficient registers are programmed correctly. Each filter is programmed into one of the two coefficient banks, and every odd coefficient must be set to zero otherwise the two interleaved data streams will corrupt each other. The table below shows how the coefficients should be programmed for the example under consideration.

CR0	a	1	0	1	0	1	0	0
	b	1	0	1	0	1	0	0
	c	1	0	1	0	1	0	0

CR1	a	1	0	1	0	1	0	0
	b	1	0	1	0	1	0	0
	c	1	0	1	0	1	0	0

9. CASCADING IMS A110s FOR INCREASED DATA PRECISION

In some high precision applications the 8 bit word length of a single IMSA110 is not sufficient. This section presents three techniques to overcome this problem. The first two combine IMSA110s with simple external hardware, the last one requires no external hardware but does place certain restrictions on the coefficients and the data.

9.1 Increasing data precision with an external 22 bit adder

The first technique makes use of an external 22 bit adder in the configuration shown in Figure 9.

At the input each 16 bit input value is split into two

8 bit words one containing the least significant 8 bits and the other containing the most significant 8 bits. Each of these 8 bit data streams is fed into an IMSA110. If the data is unsigned then both of the devices must be set to unsigned data operation. However, if the data is signed then in order to correctly process the data and preserve the sign information it is necessary for the least significant byte to be processed as unsigned data and the most significant byte to be processed as signed data (see Figure 9). This may be easily achieved by setting or clearing bit 2 of the SCR register in each IMSA110 as appropriate. The 22 bit partial results from each device are combined by making use of a 22 bit adder. This adder forms the sum of the top 14 bits (sign extended to 22 bits if signed data is being used) of the least significant partial result and the full 22 bits of the most significant partial result to give the upper 22 bits of the final result. This is combined with the lower 8 bits of the least significant partial result to give the complete 30 bit result. See Figure 10 for a graphical representation of this.

Note that for signed data, the least significant partial result must be signed extended to 30 bits as shown in Figure 9. The sign extension is easily achieved by connecting the most significant bit of the least significant partial result to the most significant 8 bits of the adder input.

This technique may be extended to give data precisions above 16 bits, however, such precisions are rarely used in practice. Sometimes it may be desired to combine a bigger filter size, as discussed in earlier sections, with increased precision. Such a system is simple to create and just involves replacing each IMSA110 in Figure 9 with the appropriate cascade of devices. Similarly multi pass filtering, as discussed in section 8, may be combined with increased precision. This is achieved by selecting a 16 bit window from the output of the system shown in Figure 9 and feeding this into the input of another high precision stage.

9.2 Increasing data precision with an external delay line

As an alternative to using an external adder it is possible to make use of the cascade adder built into each IMSA110 and an external delay line (of length D_B) as shown in Figure 11.

The rules discussed earlier in this section about signed data apply equally to this configuration. This means that if signed data was being processed then the left and right hand devices in the diagram would have to be configured for unsigned and signed operation respectively. Also CASin of device n would have to be sign extended to 22 bits as described in section 9.1. The one other consideration when increasing the data precision in this way is the number delays required in the programmable shift registers of each device.

Figure 9 : Cascade of IMSA110s for Increased Data Precision (signed data)

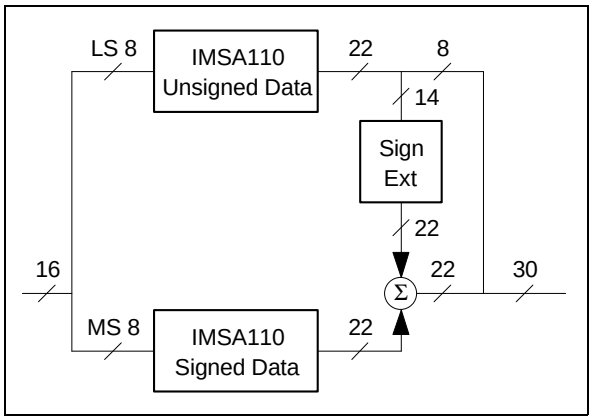


Figure 10 : Calculation of the Final Output

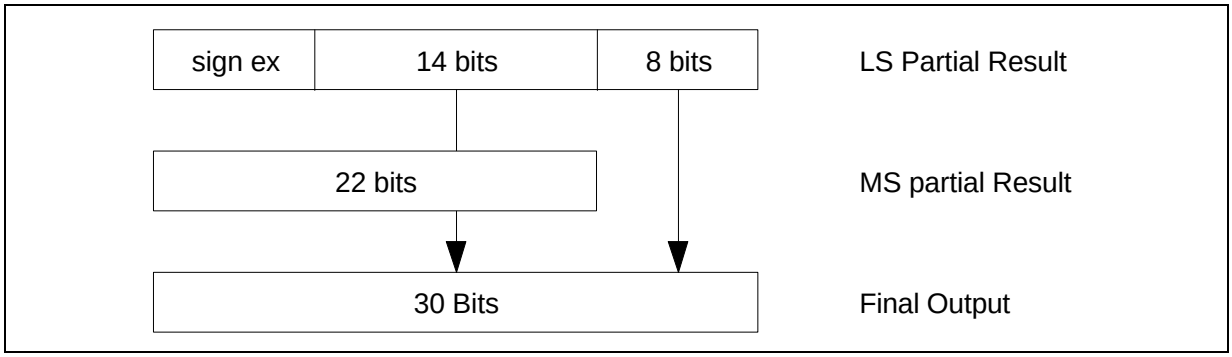
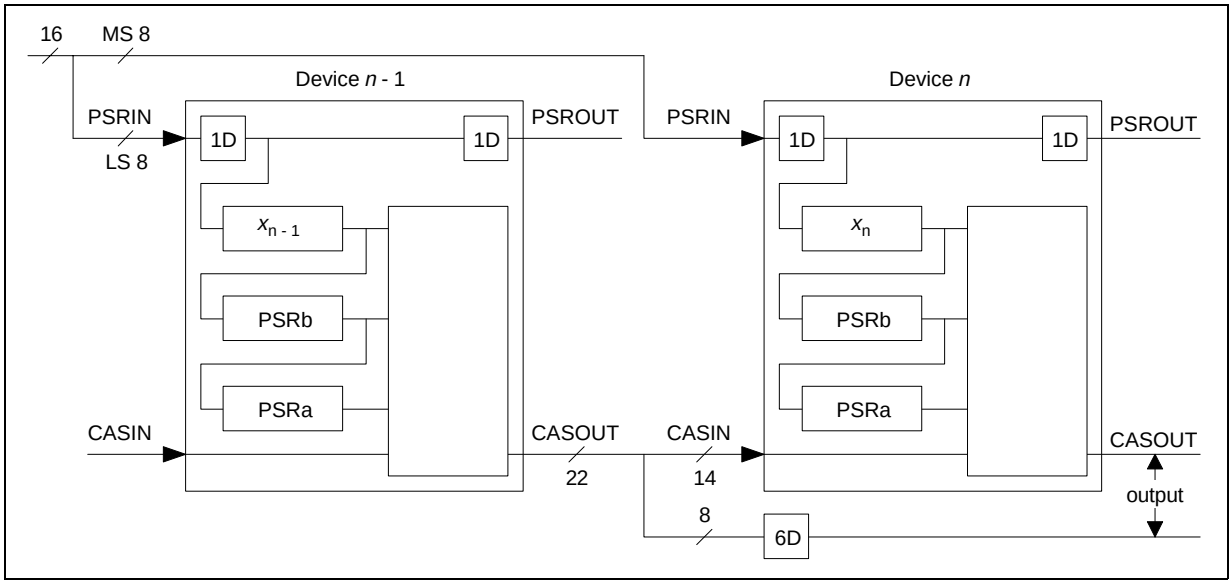


Figure 11 : Alternative Cascade of IMSA110s for Increased Data Precision (unsigned data)



Obviously the setings of PSRa and PSRb are not affected by the presence of another device and are setup as described in section 2. The setting of PSRc for each device however is important, and incorrect setting will result in erroneous calculation of the most significant 22 bits of the result.
Assuming that the delay in the PSRc of device n-1 is x_{n-1} and that the delay in PSRc of device n is x_n, it is desired to calculate the relationship between these delays for correct combination of the partial results. Consider an item of data when it reaches device n-1. The delay before the component due to this data, flowing via the reference path in device n-1, reaches the cascade adder of device n is:

$$D_{n-1} = 1+(x_{n-1}+1)+3+D_R+D_B = D_{n-1} = 41+x_n$$

Similarly the delay before the component due to this data, flowing via the reference path in device n, reaches the cascade adder of device n is:

$$D_n = 1+(x_n+1)+3+D_R = D_n = 35+x_n$$

Now for the data to be correctly aligned at the cascade adder of device n the delay along each path must be the same. Hence:

$$D_{n-1}-D_n = 0$$
$$41+x_{n-1}-35-x_n = 0$$
$$x_n = x_{n-1}+6$$

This means that the PSRc of device n must be programmed with the value which is in PSRc of device n-1 plus a fixed constant of 6.

Obviously this technique of increasing data precision may be extended beyond 16 bits, or may be combined with other cascading techniques to give larger filter sizes etc

9.3 Increasing data precision with no external hardware

If the data and coefficients are such that only 22 bits or less are required to represent the result then it is possible to increase the data precision with no external hardware. The connections required are similar to those shown in Figure 11. However, the 6 stage delay must be removed and the full 22 bits of CASout from the first device must be connected to CASin of the second device. To correctly sum the two contributions of the result, it is necessary to left shift the MAC output of the second device 8 places to the left. This shift is easily performed using the shifter in the second device, however, care must be taken to ensure that overflow does not occur. If such an overflow does occur then it will not be detected.

10. CASCADING IMSA110s FOR INCREASED COEFFICIENT PRECISION

Section 9 described three different techniques for increasing data precision by cascading IMSA110s. In this section three very similar techniques are presented for increasing coefficient precision.

10.1 Increasing coefficient precision with an external 22 bit adder

The first method makes use of an external 22 bit adder as shown in Figure 12. At the input each 8 bit value is fed to PSRin of both the IMS A110s. The

device at the top of the diagram is programmed with the least significant 8 bits of the coefficients and the device at the bottom is programmed with the most significant 8 bits of the coefficients. If the coefficients are unsigned then both of the devices must be set to unsigned coefficient operation. However, for signed coefficients, in order to correctly process the data and preserve the sign information it is necessary for unsigned and signed coefficient operation to be set in the top and bottom devices respectively (see Figure 12). This may be easily achieved by setting or clearing bit 3 of the SCR register in each IMS A110 as appropriate. Also, sign extension must be performed as described in section 9.1. The 22 bit partial results are then combined in exactly the same fashion as described in section 9.

As discussed for increased data precision this technique may be extended to more than 16 bits of accuracy if required, or may be adapted to make use of increased filter sizes etc. For very high precision systems increased coefficient and data precision may be combined to give very accurate results.

10.2 Increasing coefficient precision with an external delay line

The second method makes use of a delay line in a very similar configuration to that discussed in the previous section. A diagram showing the setup may be seen in Figure 13.

The rules discussed earlier in this section about signed coefficients still apply in this configuration. Hence if signed coefficients are required then the left and right hand devices in the diagram have to be configured for unsigned and signed coefficient operation respectively and the sign must be extended appropriately. The calculation of the setting of PSRc for each device may be calculated in the same manner as described in the previous section. When the calculation is performed the following relationship is developed:

$$subn = x_{n-1}+4$$

Figure 12 : Cascade of IMSA110s for Increased Coefficient Precision (signed coefficient)

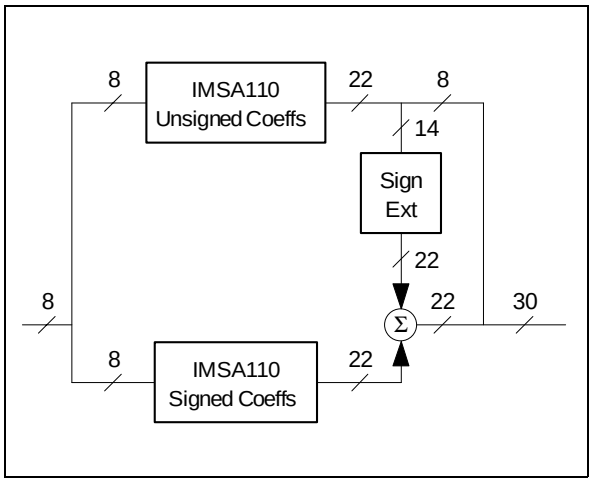
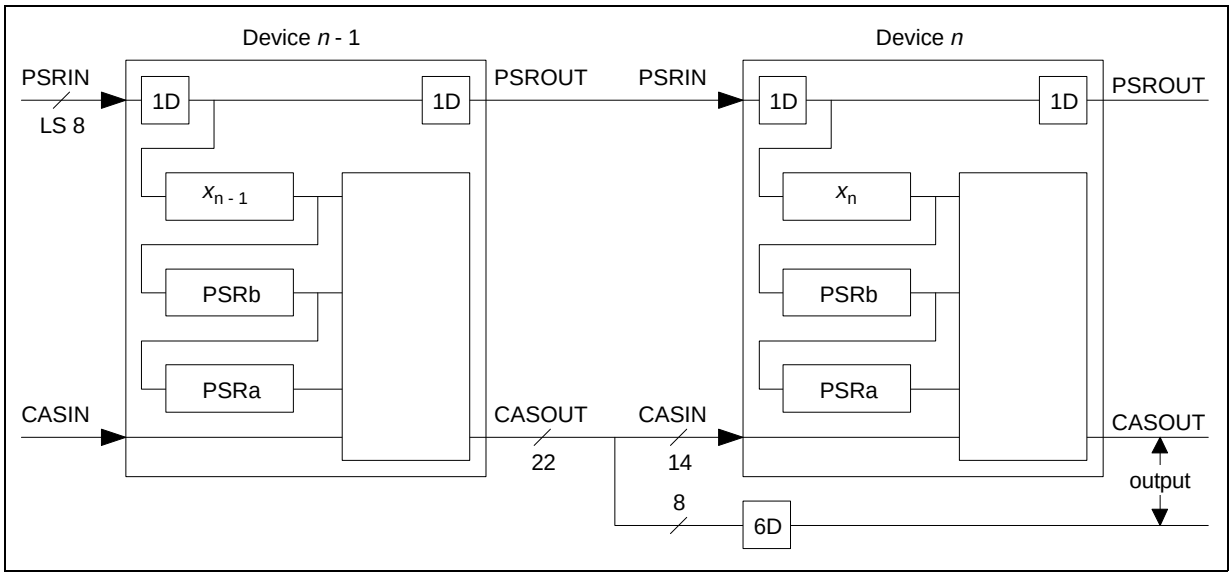


Figure 13 : Alternative Cascade of IMSA110s for Increased Coefficient Precision (unsigned coefficients)



This means that the PSRc of device n must be programmed with the value which is in PSRc of device n-1 plus a fixed constant of 4.

Obviously this technique may be extended for more precision or adapted using information presented in earlier sections to give increased filter size, multi pass filtering etc.

10.3 Increasing coefficient precision with no external hardware

It was discussed in section 9 how to achieve increased data precision without using any external hardware. Since exactly the same technique may be applied to give increased coefficient precision duplicate details are not given here.

11. SUMMARY

This document has attempted to describe some of the many ways in which IMSA110s may be cascaded to yield even higher performance. Obviously it has not been possible to discuss every possible configuration but hopefully the examples discussed should have provided both an insight into the extensive capabilities of these devices when cascaded, and some simple rules to allow easy setting up of some of the most common forms of cascades.