USN	1	P	E							

PESIT Bangalore South Campus

Hosur road, 1km before Electronic City, Bengaluru -100

Department of Electronics and Communication

INTERNAL ASSESSMENT TEST 1		
Date : 27-02-2017		Marks: **50**
Subject & Code : **DSP Algorithms and Architecture (**14ESP41**)**		Specialization: M.Tech-SP
Name of faculty : **Neethu P S**		Time : 8.30AM – 10AM

Note: Answer FIVE full questions.		**Marks**
Q.No		
1	How scaling technique is used in fixed point DSP?	**8**
	What is meant by precision of an ADC?	**2**
2	What is IIR filter? Explain the lattice structure of IIR filter.	**10**
3	Discuss the method of 3-tap data broadcast FIR filter using signal flow graph	**10**
4	Explain direct form II realization of IIR filter.	**5**
	Realize the second order system in direct form II $y(n) = -0.1y(n-1) + 0.2y(n-2) + 3x(n) + 3.6x(n-1) + 0.6x(n-2)$	**5**
5	What is rounding and explain how it affects system performance?	**10**
6	a) Realize the following system function using minimum number of multipliers. i) $H(Z) = 1 + \frac{1}{3}Z^{-1} + \frac{1}{4}Z^{-2} + \frac{1}{4}Z^{-3} + \frac{1}{3}Z^{-4} + Z^{-5}$ ii) $H(z) = \left(1 + z^{-1}\right)\left(1 + \frac{1}{2}z^{-1} + \frac{1}{2}z^{-2} + z^{-3}\right)$	**6**
	b) What is coefficient quantization in digital filters?	**4**
7	Explain how block diagrams are useful for the graphical representation of DSP algorithms.	**10**
8	Explain about data flow graphs and how it is different from signal flow graph. In data-flow graph (DFG) representations, the nodes represent computations (or functions or subtasks) and the directed edges represent data paths (communications between nodes) and each edge has a nonnegative number of	**10**

	delays associated with it. For example, Fig. 1.23(b) is a data-flow graph of the computation y{n) = ay{n - 1) + x(n), where node A represents addition and node B represents multiplication, the edge from A to B (denoted as A -*■ B) contains one delay and the edge from B to A (B -> A) contains no delay. Associated with each node is its execution time in terms of normalized time units (u.t.; units of time). For example, the execution time of node A is 2 u.t. and the execution time of node B is 4 u.t., as shown in Fig. 1.23(b). The iteration period of this DFG equals 6 u.t. The data-flow graph captures the data-driven property of DSP algorithms	

4 Explain direct form II realization of IIR filter. 5

2 mark

$$y(n) = f_0(n) = g_0(n)$$

$$= f_1(n) - k_1 g_0(n-1)$$

$$= f_2(n) - k_2 g_1(n-1) - k_1 g_0(n-1)$$

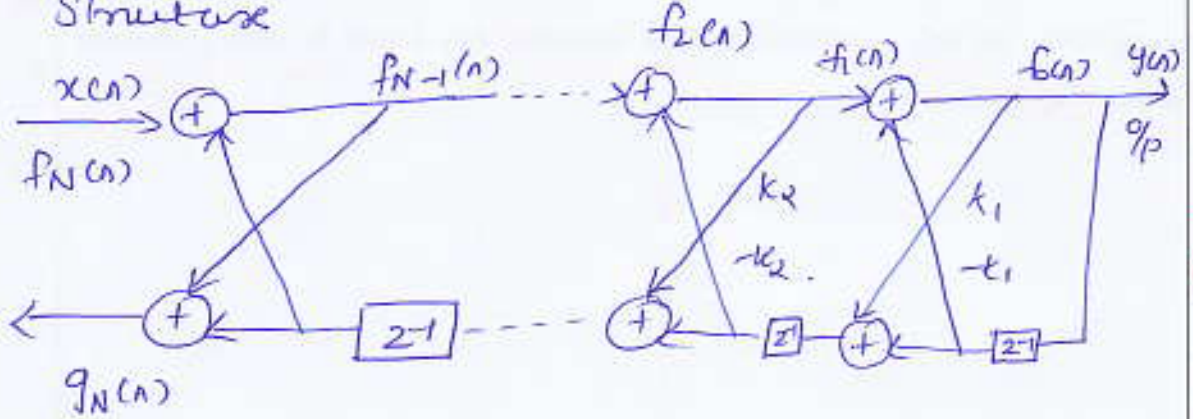
$$= f_2(n) - k_2 [k_1 f_0(n-1) + g_0(n-2)] - k_1 g_0(n-1)$$

$$= x(n) - k_1(1+k_2) y(n-1) - k_2 y(n-2)$$

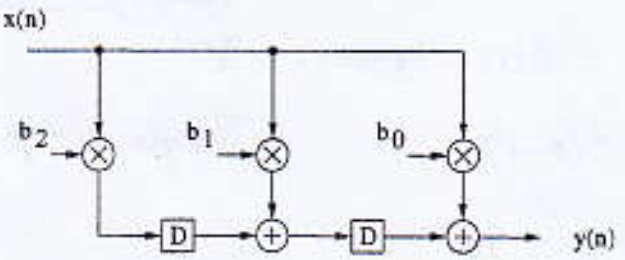
Also $g_2(n) = k_2 y(n) + k_1(1+k_2) y(n-1) + y(n-2)$

$g_2(0) = 1 \quad g_2(1) = 1 = k_1(1+k_2) \quad g_2(2) = k_2$

For a N stage IIR filter realized in lattice structure



2. (figure)

3	Discuss the method of 3-tap data broadcast FIR filter using signal flow graph A signal-flow graph (SFG) is a collection of nodes and directed edge. The nodes represent computations or tasks. A directed edge (j, k) denotes a branch originating from node j and terminating at node k. With input signal at node j and output signal at node k, the edge (j, k) denotes a linear transformation from the signal at node j to the signal at node k. SFGs have been used for the analysis, representation, and evaluation of linear digital networks, especially digital filter structures. In digital networks, the edges are usually restricted to constant gain multipliers or delay. Adders and multipliers can be described by a node with multiple incoming edges and one outgoing edge. There are 2 special types of nodes in an SFG, source nodes and sink nodes. **A source node** is a node with no entering edges, and is used to represent the injection of external inputs into a graph. **A sink node** is a node with only entering edges, and is used to extract outputs from a graph.	3
	 *Fig. 1.21* Block diagram of a 3-tap data-broadcast FIR filter.	5(FIGURES)
	The SFG of the direct-form 3-tap FIR filter where an edge with no explicit indication of operation indicates an edge with unit-gain (identity transformation). Note that the source and sink nodes in this SFG are connected to the rest of the graph through unit-gain edges in order to clearly show the input and output of the system.	

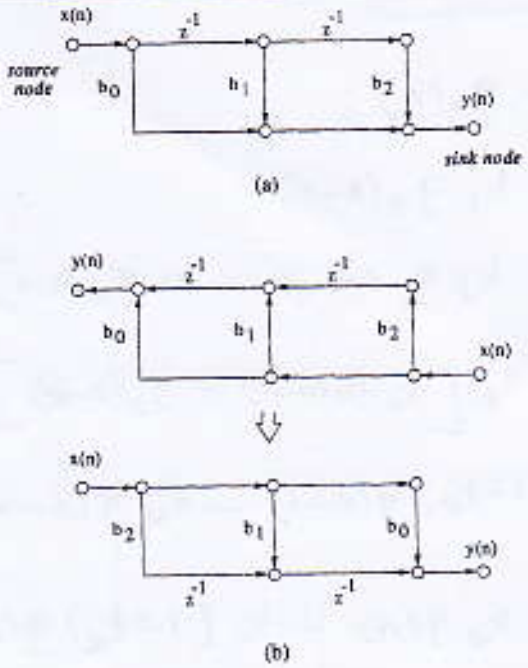


Fig. 1.22 An SFG of a 3-tap FIR filter. (a) Direct-form FIR filter SFG; (b) transposed FIR filter SFG.

Transposition of an SFG is carried out by reversing the directions of all the edges, exchanging the input and output nodes while keeping the edge gain or edge delay unchanged. The resulting SFG maintains the same system functionality. Fig. shows the derivation of the transpose FIR filters SFG (corresponding to the data- broadcast block diagram in Fig. 1.21 from the direct-form SFG.

In general, SFGs are only applicable to linear networks; they cannot be used to describe multirate DSP systems.

2

TOTAL 10

P.E.S. Istitute of Technology(Bangalore South Campus)

Hosur Road, (1Km Before Electronic City), Bangalore 560100.

Department of Electronics and Communication

SCHEME AND SOLUTION

____ INTERNAL TEST

Faculty: **Semester:**

Subject: **Sub. Code:**

Q.No.	Questions and its answers	Marks
4	Direct form II realization of IIR filter ?	5

Consider the difference eqn of the form

$$y(n) = -\sum_{k=1}^{N} a_k y(n-k) + \sum_{k=0}^{M} b_k x(n-k)$$

$$H(z) = \frac{Y(z)}{X(z)} = \frac{\sum_{k=0}^{M} b_k z^{-k}}{1 + \sum_{k=1}^{N} a_k z^{-k}}$$

Let $\frac{Y(z)}{X(z)} = \frac{Y(z)}{W(z)} \times \frac{W(z)}{X(z)}$ — 2

$$\frac{W(z)}{X(z)} = \frac{1}{1 + \sum_{k=1}^{N} a_k z^{-k}}$$

$$W(z) = X(z) - a_1 z^{-1} W(z) - a_2 z^{-2} W(z) \cdots a_N z^{-N} W(z)$$

$$\frac{Y(z)}{W(z)} = \sum_{k=0}^{M} b_k z^{-k}$$

$$Y(z) = b_0 W(z) + b_1 z^{-1} W(z) + \cdots b_M z^{-M} W(z)$$

$$w(n) = x(n) - a_1 w(n-1) - a_2 w(n-2) \cdots a_N w(n-N)$$

$$y(n) = b_0 w(n) + b_1 w(n-1) + \cdots b_M w(n-M)$$

P.E.S. Istitute of Technology(Bangalore South Campus)

Hosur Road, (1Km Before Electronic City), Bangalore 560100.

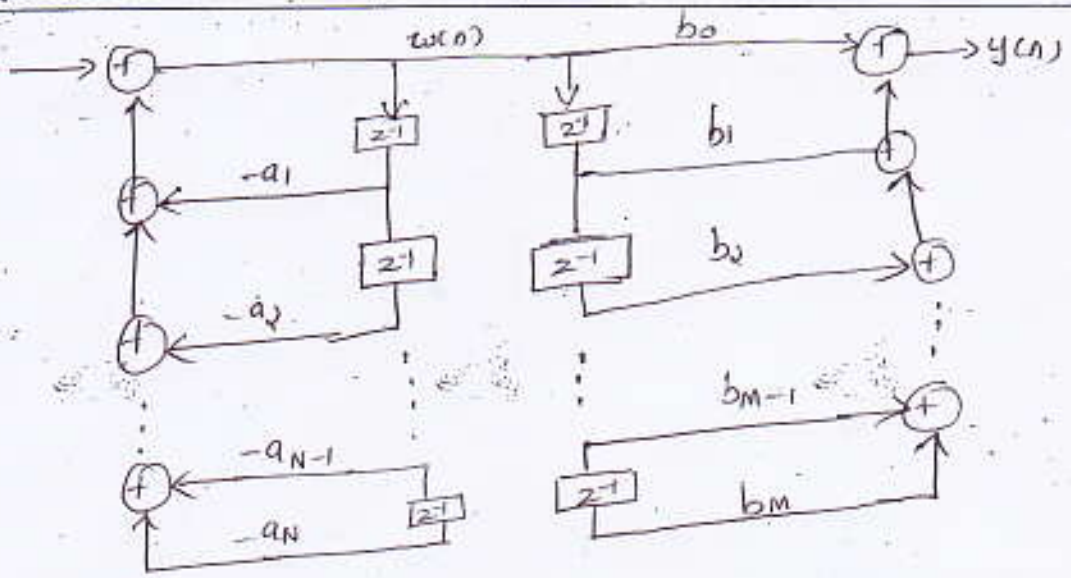
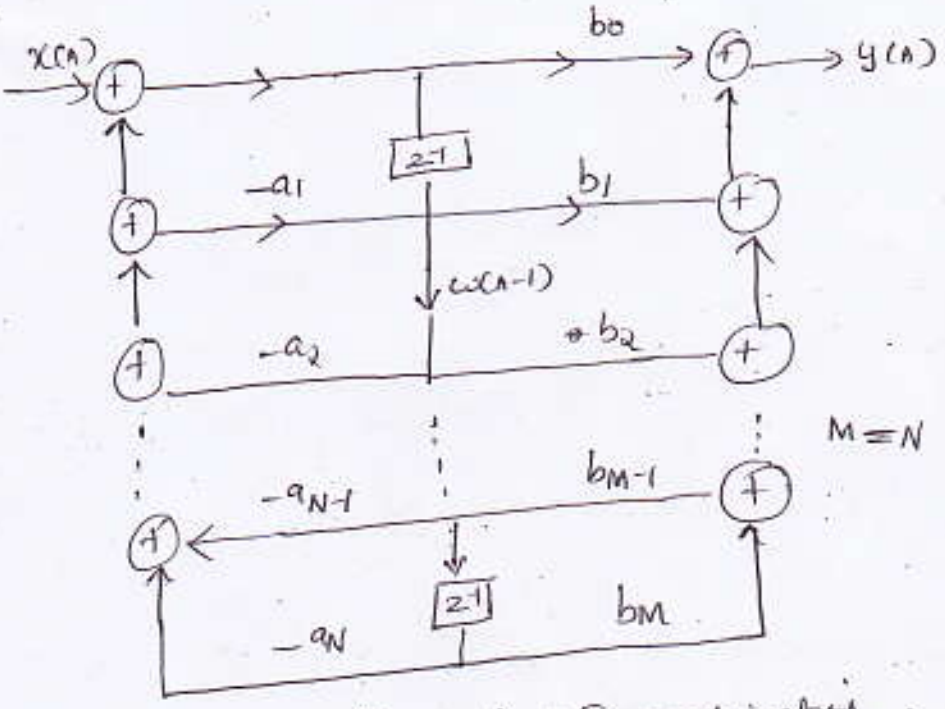
Department of Electronics and Communication

SCHEME AND SOLUTION

____ INTERNAL TEST

Faculty: **Semester:**

Subject: **Sub. Code:**

Q.No.	Questions and its answers	Marks
4		1
		2
	Direct form II realization	

Realize the second order system in direct form II

5

$y(n) = -0.1y(n-1) + 0.2y(n-2) + 3x(n) + 3.6x(n-1) + 0.6x(n-2)$

$$\frac{Y(z)}{X(z)} = \frac{3 + 3.6z^{-1} + 0.6z^{-2}}{1 + 0.1z^{-1} - 0.2z^{-2}}.$$

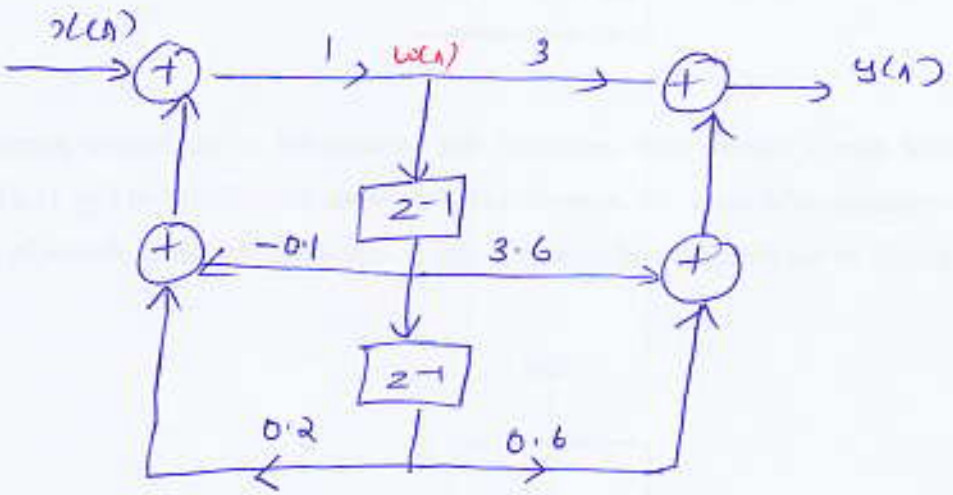
$$\frac{Y(z)}{W(z)} = 3 + 3.6z^{-1} + 0.6z^{-2}.$$

$$y(n) = 3w(n) + 3.6w(n-1) + 0.6w(n-2)$$

$$\frac{W(z)}{X(z)} = \frac{1}{1 + 0.1z^{-1} - 0.2z^{-2}}.$$

$$w(n) + 0.1w(n-1) - 0.2w(n-2) = x(n)$$

$$w(n) = x(n) - 0.1w(n-1) + 0.2w(n-2)$$



2.

3 (diagram)

5	What is rounding and explain how it affects system performance?	3

If two W-bit fixed-point fraction numbers are multiplied together, the product is (2W — 1) bits long. This product must eventually be quantized to W-bits by rounding or truncation.For example, consider the 1st-order IIR filter shown in Fig. 11.2. Assume that the input wordlength is W = 8 bits. If the multiplier coefficient wordlength is also the same, then to maintain full precision in the output we need to increase the output wordlength by 8 bits per iteration. This is clearly infeasible. The alternative is to round off (or truncate) the output to its nearest 8-bit representation. The result of such quantization introduces roundoff noise e(n).

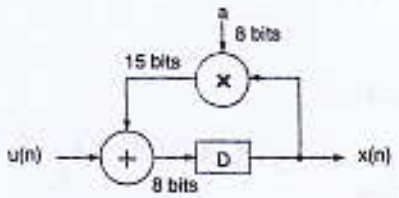


Fig. 11.2 A 1st-order IIR filter (W = 8).

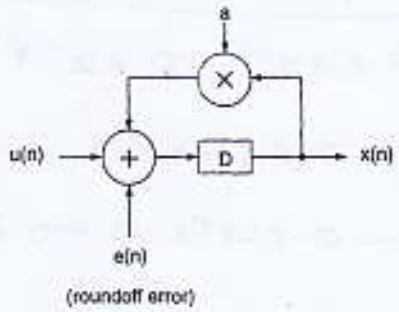


Fig. 11.3 Model of roundoff error.

For mathematical ease a system with round-off can be modeled as an infinite precision system with an external error input. For example in the previous case (shown in Fig. 11.2) we round off the output of the multiply add operation and an equivalent model is shown in Fig. 11.3.

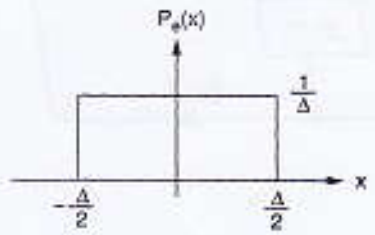


Fig. 11.4 Error probability distribution.

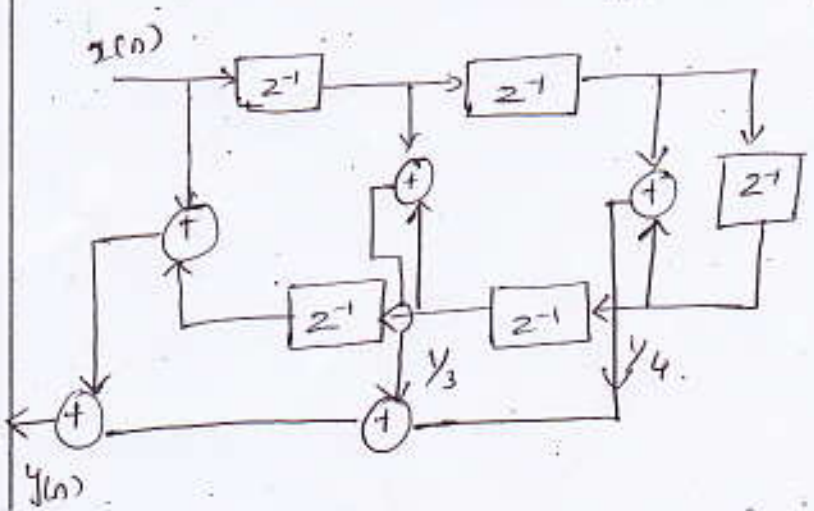
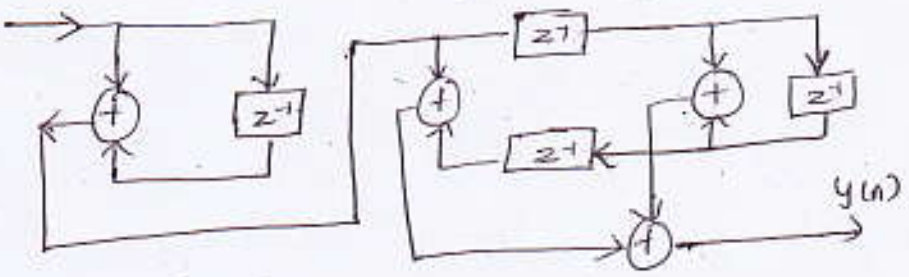
Although rounding is not a linear operation, its effect at the output can be analyzed using linear system theory with the following assumptions about e(n):

	1. e(n) is uniformly distributed white noise. 2. e(n) is a wide-sense stationary random process, i.e., mean and co- variance of e(n) are independent of the time index n. 3. e(n) is uncorrected to all other signals such as input and other noise signals. Let the word length of the output be W-bits, then the round off error e(n) can be given by $\frac{-2^{-(W-1)}}{2} \le e(n) \le \frac{2^{-(W-1)}}{2}$. (11.6) Since the error is assumed to be uniformly distributed over the interval given in (11.6), the corresponding probability distribution is shown in Fig. 11.4, where Δ is the length of the interval (i.e., $2^{-(w-1)}$ Let us compute the mean E[e(n)} and variance $E[e^2(n)]$ of this error function. $E[e(n)] = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} x P_e(x)dx = \frac{1}{\Delta}\frac{x^2}{2}\Big]_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} = 0.$ (11.7) Note that since mean is zero, variance is simply $E[e^2(n)]$. $E[e^2(n)] = \int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} x^2 P_e(x)dx = \frac{1}{\Delta}\frac{x^3}{3}\Big]_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} = \frac{\Delta^2}{12} = \frac{2^{-2W}}{3}$. (11.8) where σ_e is the variance of the round-off error in a finite precision, W-bit word-length system. Since the variance is proportional to 2^{-2W}, increase in word-length by 1 bit decreases the error by a factor of 4. The purpose of analyzing round-off noise is to determine its effect at the output signal. If the noise variance at the output is not negligible in comparison to the output signal level, the word-length should be increased or some low-noise structures should be used. Therefore, we need to compute SNR at the output, not just the noise gain to the output.	
6(a)	Realize the following system function using minimum number of multipliers. i) $H(Z) = 1 + \frac{1}{3}Z^{-1} + \frac{1}{4}Z^{-2} + \frac{1}{4}Z^{-3} + \frac{1}{3}Z^{-4} + Z^{-5}$ ii) $H(z) = \left(1 + z^{-1}\right)\left(1 + \frac{1}{2}z^{-1} + \frac{1}{2}z^{-2} + z^{-3}\right)$	6

P.E.S. Istitute of Technology(Bangalore South Campus)
Hosur Road, (1Km Before Electronic City), Bangalore 560100.
Department of Electronics and Communication
SCHEME AND SOLUTION
____ INTERNAL TEST

Faculty: **Semester:**

Subject: **Sub. Code:**

Q.No.	Questions and its answers Marks	
6	a. Realize the following S/M for using minimum no of multipliers	6
	i) $H(z) = 1 + \frac{1}{3}z^{-1} + \frac{1}{4}z^{-2} + \frac{1}{4}z^{-3} + \frac{1}{3}z^{-4} + z^{-5}$	
	$H(z)$ has linear phase symmetry property.	3
		
	ii) $H(z) = (1+z^{-1})(1+\frac{1}{2}z^{-1}+\frac{1}{2}z^{-2}+z^{-3})$	
	The given $H(z)$ is a product factors that linear phase symmetry property.	3
		

7	Explain how block diagrams are useful for the graphical representation of DSP algorithms.	10

Block diagrams are most frequently used to graphically represent DSP systems. A block diagram consists of functional blocks connected with directed edges, which represent the data flow from its input block to its output block. These edges may or may not contain delayelements. Block diagrams can be constructed for all systems with different levels of abstraction.

For example, the 3-tap FIR filter $y(n) = b_0x(n) + b_1x(n - 1) + b_2x(n - 2)$

can be described using the block diagram in Fig. 1.20, which consists of two types of functional blocks, adder and multiplier . The unit-delay elements can also be treated afunctional blocks as they are implemented using registers in reality. The unit-delay element is denoted as z^{-1} (or D)

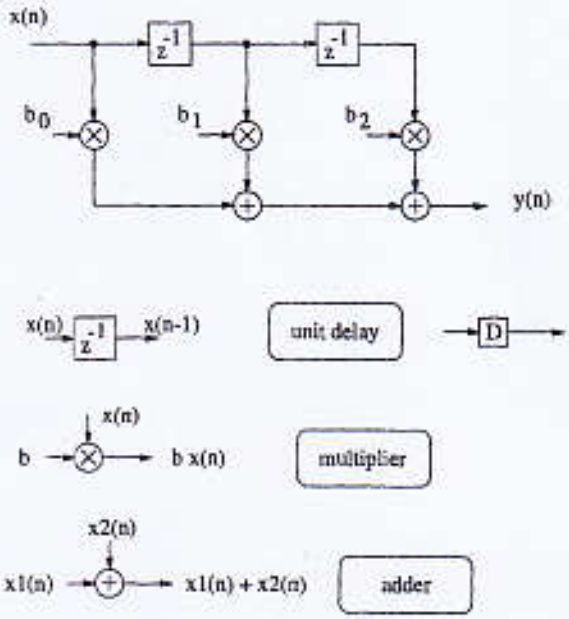
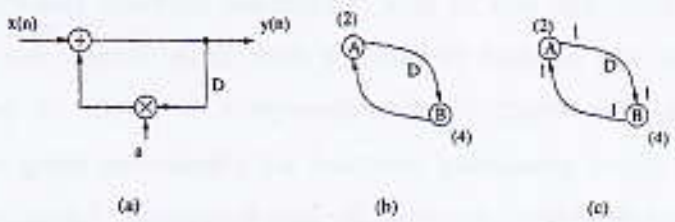


Fig. 1.20 Block diagram of a 3-tap FIR filter.

A system can be represented using various block diagrams, each of which represents a different implementation of the same functionality. For example, the 3-tap FIR filter can also be represented using the block diagram in Fig. 1.21, which is referred to as a data-broadcast structure. A block diagram is a concrete model that captures the exact functionality of a system. Each signal and functional unit are expressed explicitly in the diagram. Various block diagrams can be derived for the same system with different arrangements leading to different distinct realizations.

8. Explain about data flow graphs and how it is different from signal flow graph. (10 MKS)

In data-flow graph (DFG) representations, the **nodes represent computations** (or functions or subtasks) and the **directed edges represent data paths** (communications between nodes) and each edge has a nonnegative number of delays associated with it. For example, Fig. 1.23(b) is a data-flow graph of the computation y{n) = ay(n - 1) + x(n), where node A represents addition and node B represents multiplication, the edge from A to B (denoted as A → B)contains one delay and the edge from B to A (B→A) contains no delay. Associated with each node is its execution time in terms of normalized time units (u.t.; units of time). For example, the execution time of node A is 2 u.t. and the execution time of node B is 4 u.t., as shown in Fig. 1.23(b). The iteration period of this DFG equals 6 u.t.



2 (figure)

Fig. 1.23 (a) Block diagram description of the computation $y(n) = ay(n-1) + x(n)$. (b) Conventional DFG representation. (c) Synchronous DFG representation.

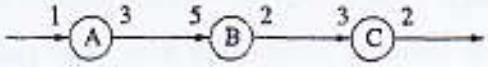
The data-flow graph captures the data-driven property of DSP algorithms. where any node can perform its computation)whenever all the input data are available. A node with multiple input edges can only fire after all its precedent nodes have fired. The latter case imposes the precedence constraints on a DFG, where each edge describes a precedence constraint between two nodes. This precedence constraint is an **intra-iteration precedence constraint** if the edge has zero delays or an **inter-iteration precedence constraint** if the edge has one or more delays. Together, the intra-iteration and inter-iteration precedence constraints specify the order in which the nodes in the DFG can be executed. 2

For example, the edge from node A to node B in Fig. 1.23(b) enforces the inter-iteration precedence constraint, which states that the execution of the k^{th} iteration of A must be completed before the $(k + 1)^{th}$ iteration of B. The edge from B to A enforces the intra-iteration precedence onstraint, which states that the k-th iteration of B must be executed before the k^{th} iteration of A.

The nodes in a DFG can be as simple as elementary indivisible operations such as addition or basic logic operations. Such DFGs are said to be **atomic**.

If the granularity is at the level of signal processing subtasks such as filtering, the DFG is a **coarse-grain data-flow graph**. A synchronous data-flow graph (SDFG) is a special case of

data-flow graph (either atomic or coarse-grain) where the number of data samples produced or consumed by each node in each execution is specified a priori. For example, Fig. 1.23(c) is a synchronous data-flow graph for the computation $y(n) = ay(n - 1) + x(n)$, which explicitly specifies that one execution of both nodes A and B consumes one data sample and produces one output. This describes a single-rate system. The SDFG can describe multirate systems in a simple way.



For example, Figure shows an SDFG representation of a multirate system, where nodes A, B, and C are operated at different frequencies fa, fb, and fc, respectively.

single- rate DFGs (SRDFGs) can also be used to represent multirate systems by DFGs, as well as block diagrams, can be used to describe both linear single- rate and nonlinear multirate DSP systems. The block diagram description is closer to actual hardware architectures where the signal processing functions are represented using functional units (blocks). The DFG, on the other hand, describes the data flow among subtasks (or elementary computations modeled as nodes) in a signal processing algorithm.

DFGs are generally used for high-level synthesis to derive concurrent implementations of DSP applications onto parallel hardware, where subtask scheduling and resource allocation are of major concerns (a schedule determines when and in which hardware units nodes can be executed