Effect of Interrupt Logic on Delay Balancing Circuit

N. Suresh Kumar
GIT, GITAM University
Visakhapatnam, India

Dr. D.V. Rama Koti Reddy
College of Engineering, Andhra University,
Visakhapatnam, India

ABSTRACT
Delay elements are added in wave-pipelined circuit to improve the performance of the circuit by reducing the delay difference of the longest and the shortest paths. But it is very difficult to obtain exact delay needed in the circuit. Instead in the present system Interrupt logic is used for delay balancing, thereby providing more feasible and accurate circuit path.

General Terms
Digital Systems, Clock Scheme.

Key words
Pipeline, Clock Skew, Interrupt controller, delay balancing.

1. INTRODUCTION
As the sampling frequencies increases beyond 100MHz, the design of low jitter, non-overlapping clock generator becomes a critical task to ensure a satisfactory performance of high speed systems [10].A Pipe line technology is used in digital system to perform multiple tasks simultaneously and toreduce the data losses in data transactions. The conventional pipeline system facing seviour problems due to improper synchronization of clock pulses. This is an universal problem in all the digital systems mostly called jitter or skew.

The data transfer rates most importantly depend on the effective clock management. In most of the digital systems the propagation of information mainly controlled on the basis of clock pulses. The pipelining includes many of latches/flip-flops. The clock is fed to every logic gate due to which the loads on clock network are unbalanced and clock skew is higher. This further reduces the effective time available to compute logic in a clock period. The penalty due to clock skew becomes worse as pipeline depths are reduced to achieve higher clock frequencies. Thus, an efficient clocking method is very critical to achieve high performance. Here a new system is implemented in the path of the clock to remove or reduce the clock skew.

In the present work a new way of clock system is proposed in the path of the clock to remove or reduce the clock skew. There are already few methods effectively working on clock skew such asMesychronous pipeline [1] and wave-pipelining [6] methods. The idea of wave-pipelining [6] was originally introduced by Cotten [7], who named it *maximum rate pipelining.*Cotton observed that the rate at which logic can propagate through the circuit depends not on the longest path delay but on the difference between the longest and the shortest path delays. As a result, several computation "waves," i.e., logic signals related to different clock cycles, can propagate through the logic simultaneously. The system clocking must be such that the output data is clocked *after* the latest data has arrived at the outputs and *before* the earliest data arrives at the outputs from the *next* clock cycle. Critical speed-limiting factors in wave-pipelining [6] are the uncontrolled clock-skew, the sampling time of registers, and the worst case transition time at the logic outputs. While the minimization of these factors has been a major challenge in the design of conventional high-speed pipelined systems as well, the equalization of path delays comes as a new challenge for the design of wave-pipelined systems. Different clock signal paths can have different delays for a variety of reasons [8]. Differences in delays of any active buffers within the clock distribution network may cause un-synchronization of data and clock in wave pipeline method.

2. EXISTING METHODS
A number of clocking techniques are proposed [11],[12] to improve the tolerance towards clock skew and toavoid latching overheads. In multi-phase clocking scheme[12] different phases of the clock are distributed to differentstages of logic. This avoids the need of pipelinelatches and provides higher skew tolerance at the costof higher clock power and increased wiring complexity.The delayed clocking scheme [11] achieves the sameby distributing a delayed pre-charge clock to the logicgates. The delayed clocking scheme is easier to implementand is scalable. Skew tolerance in delayed clockingscheme depends on the timing relation between thedelayed clocks and the global clock.

In conventional pipeline system a single clock pulse is applied to manage the data transmission through the

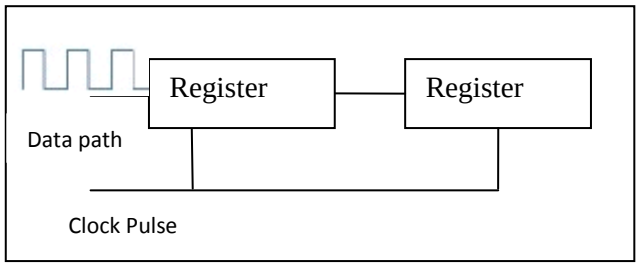

Figure 1 Conventional Pipeline System

registers in the pipeline as shown in figure 1. But it will create a clock skew in the pipeline which will decrease the data speed from one stage to other stage. The data pulses are fed into the first register when clock pulse is applied to the first stage of the pipeline. The pulse will be passed to the next stage after applying the clock pulse to the next stage. The clock pulse path is directly given to the registers where the data pulses are passes from one stage to another stage. This may create a problem of overlapping of pulses in the first stage before it enters into the next stage or it will increase the delay in data transmission. In conventional pipeline systems the clock signal is derived as

$$T_{clkconv} \geq D_{\max} + D_r + T_s + \Delta_{clk}.$$

And one of the biggest challenges in many of the clock based systems is clock pulse width. Smaller clock periods are achieved in wave pipelining [2][6] by reducing the maximum propagation delay (Dmax) by splitting the stages into number of stages.

2.1 Timing Constraints

For a wave-pipelined system to operate correctly, the systemclocking must be such that the output data is clocked afterthelatest data has arrived at the outputs and beforethe earliestdata from the *next* clock cycle arrives at the outputs. Itshall first derive the conditions for clocking of the latestand the earliest data propagating in the circuit, which is representing the *register constraints*. So introducing parameter N, whichrepresents the number of clock cycles needed for a signal topropagate through the logic block before being latched by theoutput register. This parameter serves as an intuitive measureof the degree of wave-pipelining. The data should be clockedat time T_L by the rising edge of the output register N clockcycles after it has been clocked by the input register. Due topossible constructive skew Δ (of arbitrary value) between theoutput and the input registers, this time can be expressed as

$$T_L = NT_{clk} + \Delta \text{ ---------- (1)}$$

2.2 Register Constraints

a) Clocking of the latest data: This constraint requiresthat the latest possible signal arrives early enough to beclocked by the output register during Nth clock cycle.Therefore, the lower bound onT_L , which denotes the time atwhich the output wave is captured, is given by

$$T_L \rangle D_r + D_{\max} + T_s + \Delta_{clk} \text{ ---------- (2)}$$

*b) Clocking of the earliest data:*This condition requiresthat the arrival of the *next* wave i+1 must not interfere with theclocking of the current wave. That is, the earliest possiblesignal of wave must arrive later than the clocking of thewave at the output register. This condition is similar to the*race-through* constraint in conventional pipelining. Notice thatthe earliest arrival of wave i+1 is given by $T_{clk} + D_r + D_{\min}$.

After the clock pulse has been applied to the output register,additional hold time$T_h$must be allowed for the data to remainsteady. In addition, one must account for an uncontrolledclock-skew Δ_{clk} at the output register. As a result, theis bounded above as follows:

$$T_L < T_{clk} + D_r + D_{\min} - (\Delta_{clk} + T_h) \text{ ---------- (3)}$$

Combining constraints (2) and (3) gives us the well-known*maximum rate pipelining* condition of Cotten

$$T_{clk} > (D_{\max} - D_{\min}) + T_s + T_h + 2\Delta_{clk} \text{ ---------- (4)}$$

The minimum clock period is limited by the difference inpath delays(D_{max} - D_{min}), plus the *clocking overhead*($Ts + T_h + 2\Delta_{clk}$) resulting from the insertion of clockedregisters.So the clock signal is derived in the wave pipelining is

$$T_{clk.w} \geq (D_{\max} - D_{\min}) + T_h + T_s + 2\Delta_{clk}$$

The wave pipelining is shown in figure 2. The maximum performance of a wave-pipelined circuit is limited by the delay differences of the longest and shortestpaths in the circuit, the setup(Ts) and hold time(Th) of the storage elements between the pipeline stages, the clock skew, and process variations. One obvious way to improve cycle time isto attack the delay balancingproblem, i.e. reduce the delaydifference between the longest and shortest paths. Better balance permits faster clocks and more simultaneously activewave in the circuit.

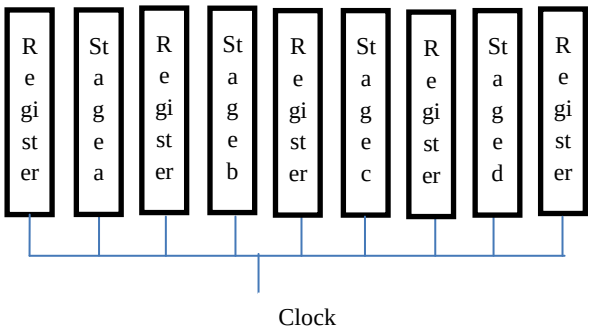

Figure 2 Wave Pipelining

Andfurther the propagation delay is reduced and the clock synchronization is controlled by introducing a delay element in the path of clock signal of Me-synchronous pipelining [1] s shown in figure 3. This delay will be equal to the delay created by the pulse passed from one stage to other stage of the pipeline. The system is clocked such that a pipeline stage is operating on more than one data wave simultaneously. At any given time, multiple waves can be present in a stage and the waves are separated based on physical properties of internal nodes in the logic stage. The clock signal is derived in the Mesynchronous pipelining is

$$T_{clk.m} \geq (D_{\max}(f) - D_{\min}(f)) + T_h + T_s + 2\Delta_{clk}$$

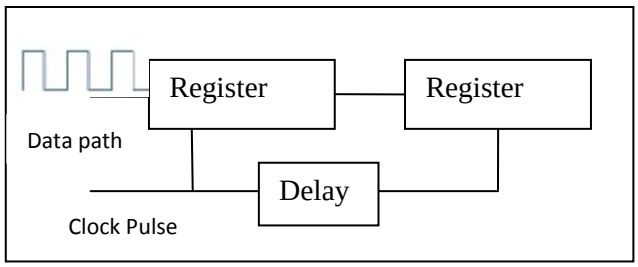

Figure 3 Mesynchronous pipelining

Most of the delay balancing techniques use delay elements either as padding in the data path of the circuit or to insert skew in some clock distribution lines of the circuit. But, it is not easy to design an exact delay needed [9]. Furthermore, fora circuit with large delay differences, a series of delay elementsneeded to insert to balance the paths. It is even harder to achieve the exact delay value desired.

In the present method Interrupt logic is used to control delay which is more accurate than using long chains of delay elements because the clock lines which control latches, in contrast to data signals, have smaller and more easily controllable skews.

2.3Clock Edges Requirements

The inputs are defined either at rising or falling edges of the sampling clock. So the clocking edges directly imply the inaccuracy would be related to timing jitter. So it is highly essential to achieve non overlap timing signals to get accurate signals. On the other hand in most of the applications with higher accuracy the rising edge of pre-phase could be placed slightly earlier than that of post phases to further suppress the charge injection errors.

3. ENHANCED METHOD

In multiple stages pipeline system different delay elements need to be inserting in the clock path. Because the first delay inserted in the clock path is equal to the data propagated from first stage to second stage. And the second delay element is such that the sum of delay1 and delay2 must be equal to delay generated in data propagating from stage 1 to stage 3. Although the data propagates immediately from stage 2 to stage 3, the stage 2 data will be fed tostage 3 only when the stage 1 data enter into stage 2. The stage 2 data is pushed into stage 3 when stage 1 data enters into stage 2. In the present work a new method is proposed by introducing a controller to supply the clock pulses to thepipeline stages instead of using external clock circuitry as shown in figure 4. The individual stage in the pipeline interrupts the interrupt controller to enable the next stage clock pulse for the pipeline[14].

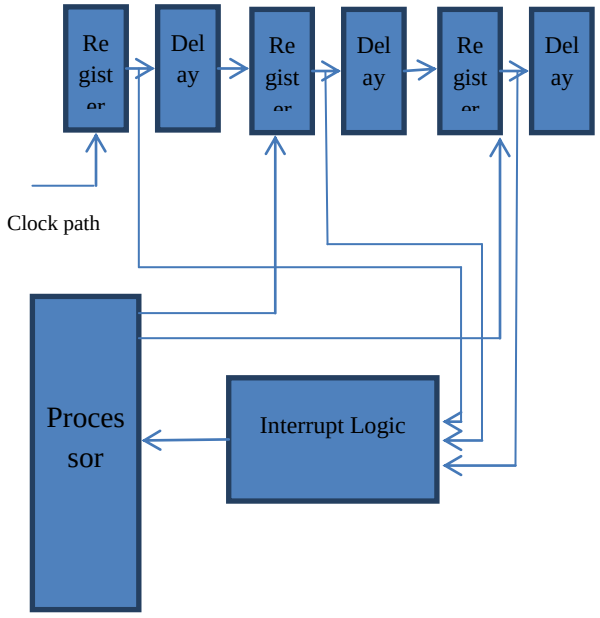

Figure 4 Block diagram of Enhanced clock system in pipe line system

The system is clocked such that a pipeline stage is operating on more than one pulse simultaneously. The presentsystem at any given time, multiple pulses can be present in a stage similar to the mesychronous pipelining.As one pulse enters into the first flip-flop it sets the next flip-flop in the present stage. While the first pulse entering into the next FF next pulse simultaneously operates the first FF. The clock signal in the present method is equal to

$$T_{clk.i} \geq (D_{max} - D_{min}) + T_h + T_s + 2\Delta_{clk}$$

where, T_h = holding time of the register
T_s = Setting time of the register
D_{max} = maximum propagation delay
D_{min} = minimum propagation delay
Δ_{clk} = Clock Uncertainty

So the propagation delay will be less than the total clock periodof the pipeline stages.

For *balancing* the minimum and maximum *delays* of a circuit path, interrupt logics are preferred over delay elements. Since delay elements with Min/Max delays do not reduce the delay difference of such a path. If the Min/Max delay ratio of a delay element is less than 1, the delay difference of such a path always increases with inserted delay elements. The delay obtained at each clock which is applied at each stage of the circuit equal to the propagation delay of the wave from one stage to next stage through register in the present stage and fixed fractional delay element inserted between two stages.

3.1 Fractional Delay

Due to the dynamically changing nature of the delays, typicallysmoothly changing from 10 ms to about 50 ms, we need to implementfractional delay lines. If we were to use non-fractional delays, we wouldonly be able to implement step-wise delays dictated by the sampling ratefs and integer K as seen in Eq. (3.1.1). This means that each delay incrementor decrement would be confined to integer units of T = 1/fs seconds.

$$Delay_{Non.fract} = K.\frac{1}{f_s} \quad \text{---------- (3.1.1)}$$

Linear interpolation of delay is quite straightforward to implement andis shown in Eq. (2) where τ is the fractional delay amount, $y_{fd}[n-\tau]$ isthe delayed output sample value, frac(τ) is the fractional part (mantissa),and int(τ) is the integer part of the fractional delay τ.

Figure 5 Single Block Delay Unit

$$y_{fd}[n-\tau] = a_{int(\tau)}.y[n-int(\tau)] + a_{int(\tau)-1}.y[n-int(\tau)-1]$$

---------- (3.1.2)

$$a_{int(\tau)} = 1 - frac(\tau)$$

$$a_{int(\tau)-1} = frac(\tau) \quad \text{---------- (3.1.3)}$$

$$a_{int(\tau)} + a_{int(\tau)-1} = 1 \quad \text{---------- (3.1.4)}$$

Note that the resulting fractional delay in essence is just a linear weightingof two adjacent delayed samples. For example, if the fractional delay wewant is 6.5, the weights $a_{int}(\tau)$ and $a_{int}(\tau)-1$ will be both equal to 0.5, whilethe integer delays equal to 6 and 7 respectively:

$$y_{fd}[n-\tau]|_{\tau=6.5} = 0.5.y[n-6] + 0.5.y[n-7]$$

a dynamic fractional delay line.

$$y_{fd}[n-\tau] = a_0.y[n] + a_1.y[n-1] \text{ ---------- (5)}$$

$$a_0 = 1 - frac(\tau) \quad \text{---------- (6)}$$

$$a_1 = frac(\tau) \quad \text{---------- (7)}$$

3.2 Clock Scheme

A key element of most digital systems is the clock. Its period determines the rate at which data are processed, and so should be made as small as possible, consistent with reliable operation.

Based on a worst case analysis, clocking schemes for high performance systems are analysed. These are 1- and 2-phase systems using simple clocked latches.

When traditional registers are used, single-phase clocking is neither safe nor fast [13] and a multi-phase clock is required. This is due to lower bound constraint on the minimum short path delay of the combinational circuits that could result in a data racethrough problem. In the present work a microcontroller used to generate a clock pulse from the global clock. The clock period can be varied depends on the signal arrived from Interrupt logic. And the clock pulse comes $T_{\mu cf}$after the rising edge on the global clock. This clock pulse is then distributed to different stages of pipeline.

After arriving of first valid data at interrupt controller the interrupt controller interrupts microcontroller. In response to this interrupt the microcontroller send a clock signal to the next stage register. Similarly after receiving a valid signal from second register the interrupt controller again interrupts the microcontroller. The microcontroller in the same way activates the next stage in a different path.

The main goal in the most of the digital systems to design a clocking scheme is to make the period as small as possible,this is to maximizing the speed of the system. It is obvious that minimizing D_{max} is basic to minimizing theclock period. But, as pointed out above, it-is also important tokeep the smallest path delay D_{min} as large as possible. But it is not that much easy to make the logic path delays uniform in value.For this reason, a system design is proposed here to control the clock skew at multiphase through a programmable controller. And it is very difficult to manage the clock system between the pipeline stages and delay elements by satisfying all the 2-phase constraints [13].

4. HARDWARE

The hardware circuit description is shown in figure 6.Initially the data pulses are fed to the first stage of the pipeline. In the first stage the first bit enters into first position.When the data pulse feeds the next pulse the first bit pushed to the next position and the first position will bereplaced with the new bit. The bit positions will change in queue. When the last bit overflowed from the first registerand it will enter into the next stage. At the same time the overflow bit from the first register send to interrupt requestpin (IRQ) of the Programmable Interrupt Controller (PIC). The PIC will generate an interrupt request tomicrocontroller (8051) on one of the Port pin. When the first bit of Interrupt Request register is set after IRQ0 receives a request, a signal willbe generated through a Portpin as clock pulse to the second stage. Immediately after clock signal arrived at second stage,the overflowed bit from first stage enters into second stage through a delay element. An individual port pin is used for clock signal for first stage. Theseclock signals are controlled and change it states through microcontroller programming. The PIC is interfaced andcontrolled through programming in microcontroller. The initialization command word of PIC is set to Edge Trigger Mode in order to achieve faster response from pipeline stages. As the overflow occurs from second stage the overflow bit willtry to enter into next stage. At the same time there may be a chance of overflow from first stage to second stage. Soin the case individual clock pulses need to be set in Port 1. So p1.1 and p1.2 will generate clock signals for secondand third stage. Here the width of the pulse generated from the port pins($T_{\mu pcw.i}$) must be greater than the propagation delay of data from one register to other register. The propagation delay ($D_{prop\_delay} = D_{min} - D_{max}$)between any two registers depends on holding and setting time of the internal wave through an individual register and small fractional delay element (Y_{fd}).

i.,e., $$D_{\min} - D_{\max} \geq T_h + T_s + 2\Delta_{clk} + y_{fd}$$

$$T_{\mu pcw.i} \geq D_{prop\_delay}$$

The amount of Y_{fd}indirectly depends on the addition of IRQ response (R_{IRQ}) and INTR response (T_{INTR}) at microcontroller $R_{IRQ} + T_{INTR}$. Because this sum creates an addition delay in the clock path generated by ports. This additional added delay balanced and cancelled by attaching small fractional delay between two stages. So here the delay produced by $R_{IRQ} + T_{INTR}$ is balanced by $Y_{fd.}$

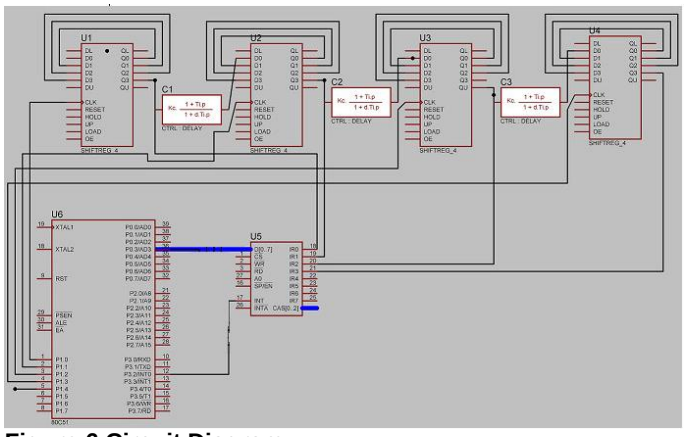

Figure 6 Circuit Diagram

In the same way the pulses will be supplied when overflow occur at higher stages. The system is clocked such that a pipeline stage is operating on more than one pulse simultaneously.

5. EXPERIMENTAL RESULTS

The data transmission through different stages is controlled by small interrupt logic in fixed priority mode. The respective interrupt requests will control the clock pulses of respective stages of pipeline. The requests for the next stages stored in PIC are processed by microcontroller and hence individual clock signals are managed. The pipe line operations are simulated and understand by CAD tools. The hold time of the registers is reduced so it reasons high performance. The time period in obtaining Q after clock is ascertained is minimized.

Figure 7 Analysis of Pipe Line using Timing Diagram

The widths of individual clock pulses to different stages are independent and hence can be maintained at short width. Therefore higher performance is achieved in terms of data transfer rates. The relative error at Q output is minimized to almost zero. Pipeline constructed with fewer flip flops and hence few registers are involved in the design. Higher data rates are observed in the current pipeline system. Smaller clock periods are achieved using interrupt logic. Higher performance

can be achieved using simple control of clock distribution. Fetching waves and processing the output are simultaneously performed at different speeds by varying the input clock frequency.

6. CONCLUSION

The clock distribution becomes simpler by controlling clock signals by internal ports of controller. Simplesoftware logic is used to control the ports of controller to generate the clock signals. Higher performance can beachieved using interrupt based delay balanced mode. The proposed pipeline scheme avoids wave collision and supports high accuracy in data transmission. In the proposed system the clock speed mainly affected by interrupt controller. It avoids predicting the delay elements in the clock path. One interruptcontroller must be dedicating for generating interrupts. Controlling of Initializing Command Words and OperationalCommand Words increases the complexity in the code.

7. REFERENCES

[1] Suryanarayana B. Tatapudi, Student Member, IEEE and José G. Delgado-Frias, Senior Member, IEEE,A Mesychronous high performancedigital systems, VOL. 53, NO. 5, MAY 2006.

[2] C.Thomas gay, "Timing constraints for wave pipelinedsystems" IEEE transactions on Computer aided design of integrated circuits, vol13,no.8, august 1994.

[3] Jabulani Nyathi, "A high performance hybrid wave pipelined linear feedback shift register with skew tolerant clocks", IEEE, 1384- 1387,2004.

[4] Mohammad Maymandi, "A digital programmable delay element: Design and analysis", IEEE transaction VLSI systems, Vol.11, no.5,October 2003.

[5] David E. Duarte, "A Clock Power Model to Evaluate Impact of Architectural and Technology Optimizations", IEEE transactions on very large scale integration (VLSI) systems, vol. 10, no. 6, December 2002.

[6] Wayne P. Burleson, "Wave-Pipelining: A Tutorial and Research Survey", IEEE transactions on very large scale integration (VLSI) systems, vol. 6, no. 3, September 1998.

[7] L. Cotten, "Maximum rate pipelined systems," in Proc. AFIPS Spring Joint Comput. Conf., 1969.

[8] EBY G. Friedman, "Clock Distribution Networks in Synchronous Digital Integrated Circuits**",** Invited paper,proceedings of the IEEE, VOL. 89, NO. 5, May 2001 665.

[9] S. H. Unger, C. J. Tan, "Clocking schemes for high-speedDigital systems," IEEE Trans. on Computers, vol. C-35,No. 10, Oct. 1986, pp. 880-895.

[10] Sai.Weng Sin, "Novel Timing skew insensitive, Multiphase clock generation scheme for parallel DAC and N-path filter", RIUPEEEC 2006, pp133-136

[11] Silberman et al., "A 1.0-GHz Single-Issue 64-Bit PowerPCInteger processor," IEEE Journal of Solid State Circuits,vol. 33, Nov. 1998, pp. 1600-07.

[12] D. Harris et al., "Skew-Tolerant Domino circuits" ISSCCDigest of Technical Papers, Feb 1997, pp. 416-417.

[13] Stephen h. unger,"Clocking Schemes for High-Speed Digital Systems", IEEE Transactions on computers, vol. c-35, no. 10, October 1986, pp880-895.

[14] Nandigam.S, "A New Method to Enhance Performance of Digital Frequency Measurement and Minimize the Clock Skew", accepted to publish in the future issue of IEEE Sensor J.

[15] T.N. Prabakar, "Design and implementation of an Asynchronous Controller for FPGA Based Asynchronous Systems", 2010International Journal of Computer Applications (0975 - 8887) Volume 1 – No. 21, pp 23- 29.