$\begin{aligned} \forall (x) &= \begin{cases} 0, & x < 0, \\ \forall_{0}, & x \geq 0, \end{cases} \quad \sigma_{x} \sigma_{p} \geq \frac{\hbar}{2} \quad \underbrace{\eta_{1}}_{1} \eta_{0} \circ \int_{0}^{0} \underbrace{\overline{E} = h \nu}_{2} \\ \psi_{1}(x) &= \frac{i}{\sqrt{k_{1}}} (A_{+} e^{ik_{1}x_{1}} + A_{+} e^{-ik_{1}x_{1}}) \quad x < 0 \quad \underbrace{\circ \circ \circ \circ \circ \circ}_{0} \circ \underbrace{d}_{0} \underbrace{f_{+}}_{1} (A_{+}) \approx \underbrace{c}_{1} [H, A(t)] + \frac{\partial A(t)}{\partial t} \\ \psi_{2}(x) &= \frac{i}{\sqrt{k_{2}}} (B_{+} e^{ik_{2}x_{1}} + B_{+} e^{-ik_{2}x_{1}}) \quad x > 0 \quad T \mid j, m \rangle \equiv |T(j,m)\rangle = (-1)^{j-m} \mid j, -m \rangle. \end{aligned}$ $\frac{i\hbar\frac{\partial}{\partial t}\Psi(\mathbf{r},t)=\hat{H}\Psi(\mathbf{r},t)}{P[a \le X \le b]=\int \int W(x,p)dpdx} H_n(x)=(-1)^n e^{x^2} \frac{d}{dx^n}(e^{-x^2}) \xrightarrow{i_1m_1}{i_2m_2} = \frac{i_2m_2}{i_3m_3} \text{ Dynamic Partial}$ $\begin{array}{c} \begin{array}{c} & \psi_{(x)=Ae^{ikx}+Be^{-ikx}} \\ & \psi_{(x)=Ae^{ikx}+Be^{-ikx}+Be^{-ikx}} \\ & \psi_{(x)=Ae^{ikx}+Be^{-ikx}+Be^{-ikx}+Be^{-ikx}} \\ & \psi_{(x)=Ae^{ikx}+Be^{-ikx}+Be^{$ $\begin{array}{c} A_{*} \bigvee \bigvee & B_{*} \circ O \\ \downarrow \circ \bigcup & A_{*} & A_{*}$ Calculator Module

By Jeanne Beau and Zachary Martin

A DPR calculator design with a serial monitor based user interface to achieve a scalable arithmetic logic unit at a consistent size This system will:
Receive a selected operator and two operands from the user

- Reconfigure itself to perform the operation
- Display the result of the operation on a serial terminal.

High-Level Architecture



AXI4-Lite Datapath & State



Software Control Logic





ALU Component - Division

Name	Value	120 ns 130 ns	140 ns 150 ns 160 ns 170 ns	180 ns 190 ns 200 ns 210 ns	220 ns 230 ns 240 ns			
V[31:	2	-1	-1015300550	-385580685	-1847796063			
V[31:	2	-1	1033192742	672184422	894796057			
→ ष Z[63:	0000001	000000100000000	0000000c37bbe3a	00000000e9048173	fffffffefc87e0d3			
谒 clock	0							
1qO 😺	/				/			



Name	Value	300 ns	310 ns	320	0 ns	330 ns	340 ns	350 ns	s 36	50 ns	370 ns	380 ns	390 ns	400 ns	410 ns	420 ns	430 ns	440 ns	450 ns	460 ns	470 ns	480 ns	490 ns
🔓 s00_axi_aclk	0																						
🔓 s00_axi_aresetn	1																						
→ 😼 s00_axi_awaddr[3:0]	4																						
→ 😼 s00_axi_awprot[2:0]	0	•																					
🔓 s00_axi_awvalid	1																						
🕨 🕏 s00_axi_wdata[31:0]	00000042	fffffff 0000000					fffffff 0000000												ffffffee 0000000				
→ 😼 s00_axi_wstrb[3:0]	f	\neq																					
🔓 s00_axi_wvalid	1																						
🔓 s00_axi_bready	0																						
🕨 🕏 s00_axi_araddr[3:0]	0						(8		0	с						0
🕨 😽 s00_axi_arprot[2:0]	0														. (ָּ 							
🔓 s00_axi_arvalid	0																						
🔓 s00_axi_rready	0																						
🔓 s00_axi_awready	1																						
🔓 s00_axi_wready	1																						
• • s00_axi_bresp[1:0]	0	0																					
🔓 s00_axi_bvalid	0																						
🔓 s00_axi_arready	0																						
🗞 s00_axi_rdata[31:0]	ffffffe									00	000000											ffi	ffffe
→ 😼 s00_axi_rresp[1:0]	0														() 							
🔓 s00_axi_rvalid	0																						
UC_S00_AXTA_WIDTH	32	32																					
UC_S00_AXDR_WIDT	4														4								
Ът	10000 ps														1000	0 ps							



Revelations & Conclusion

- Design Flow floor plan constraints
 - 1st iteration created fplan with division RM and multiplication needed DSP48 cells
 - 2nd iteration created fplan with multiplication and division needed a larger size (more carry cells)
 - Ended up drawing a much larger block which included DSP48 cells
 - This might not be ideal because the hardware requirements are significantly varied for multiplication and division.
- Division operation limitations
 - Division results were inconsistent in software testing. This was previously simulated successfully in
 Vivado.
 - Results were correct when the data was written a second time. We now do this as a work around.
 - We speculate that the number of clock cycles needed is unpredictable, not given enough time to complete the computation by the time we're ready to read the data.
 - The circuit is mostly combinational and we may be able to solve this with a more synchronous design, as discussed in a previous lecture about less clock cycles vs. combinational timing constraints.

 $\lim_{k \to 0} = a_{k} - a_{k} + \frac{1}{3} \cdot (n - 1) = -4r + \frac{1}{3} \cdot a_{k} = a_{k} = 43 + \frac{1}{3}r + 5 \cdot 5 \cdot \frac{1}{3} \cdot -43 + \frac{1}{3}r + 5 \cdot 5 \cdot \frac{1}{3} - 2r + 2 - 5 \cdot \frac{1}{3}r + \frac$ $S = \int_{1}^{2} \frac{1}{2} \int_{1}^$ 34 3 f(4) = 2 + x - 31 4. g = 0.4(1) (x Questions? court 8(0) R × X = 4 c N 4x⁴ mck 35 + (1-2) Alsch-A f = (Ab) C D = (AD) 35 + (2-5) = 2 + (2-5) + (2-5) = (2-5) + (2 V=1532x 5=12:25(x-2)+x-2/4.(2x+3) +6 (2x+3)?(x-2)5 (x) +x-2-3-x-2-1+2-3=(4)x3-1=2f(x)