Binary Divider

\[
\begin{array}{ll}
\text{divisor} & 1101 \\
\text{dividend} & 10000111 \\
\end{array}
\]

\[
\begin{array}{ll}
\text{quotient} & 1010 \\
\text{remainder} & \text{Unsigned division}
\end{array}
\]

\[
\begin{array}{ll}
135 \div 13 = 10 \text{\_remainder \_5} \\
01111 \\
\downarrow \\
-1101 \\
00101 \\
\downarrow \\
-0000 \\
0101 \text{ remainder}
\end{array}
\]

Notice this is just the opposite of multiplication.

We need to subtract and shift to the right. (But instead of shifting divisor to right we shift dividend to left.)

```
8-bit divide by 4-bit to yield 4-bit
```

```
Extra location to hold the end bit before shifting (Need for subtraction)
```
Since the divisor is larger than the 4 bits of dividend above it the first bit of the quotient will technically be a \( \emptyset \).

* Note that if we could do a subtraction here (divisor smaller than dividend) we would place a 1 in the quotient, but this would be bit 4 of the quotient which is lost \( \Rightarrow \) Overflow

In other words the divisor is small enough that the quotient requires more bits than we have reserved space for.

Example: \[ 0101 \div 0010 \quad \begin{array}{c} 11110 \hline 10010111 \\ -0101 \downarrow \\ 010000 \\ -0101 \\ 00111 \\ -0101 \\ 0101 \\ -0101 \\ 00001 \end{array} \Rightarrow 5 \overline{151} \]
However, since it doesn’t fit we shift left

\[ \begin{array}{cccccccc}
1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\
\hline
& & 1 & 1 & 0 & 1
\end{array} \]

This will be the quotient digit, but for now just put 0

Can we subtract? Yes, so assert “Su” signal.

This fills the quotient with a 1 as well as takes the difference sitting in the subtractor and loads it in the ACC register.

\[ \begin{array}{cccccccc}
0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\
\hline
\text{dividend} & \text{quotient}
\end{array} \]

Now shift left to look at next column

\[ \begin{array}{cccccccc}
0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\
\hline
1 & 1 & 0 & 1
\end{array} \]

Can we subtract? No, since divisor is too big we leave Su=0

Just shift left
Can we subtract? Yes, thus $S_n = 1$. Load subtractor result and flip right-most bit.

Now shift left to do next column.

Can we subtract? No, $S_n = 0$. We are done shifting.

Final register contents

$135 \div 13 = 10$ remainder 5
The state graph for the controller for this specific implementation is: