AES

McAlpin
CECS-UCF
AES Background

- **Advanced Encryption Standard (AES)**
  - Published in 2001, standardized in 2002.
  - AES is based on **Rijndael** cipher structure (not Feistel)
    - **Rijndael structure uses advanced mathematics (group, ring, and field theory) which we will not cover**
  - **Key size**: 128, 192 or 256 bits
  - **Block size**: 128 bits
  - Rounds (10) – similar to DES
The 128-bit version of AES uses 10 rounds to encrypt each block of the input plaintext.

Each round performs an invertible transformation on a 128-bit array, arranged as a 4-byte by 4-byte square array called the state.

The initial state $X_0$ is the XOR of the plaintext $P$ with the key $K$: $X_0 = P \oplus K$.

Round $i$ ($i = 1, \ldots, 10$) receives state $X_{i-1}$ as input and produces state $X_i$.

The ciphertext $C$ (for the block) is the output of the final round: $C = X_{10}$.
AES Round Processing

• *In each round*, the state undergoes:
  
  • **SubBytes step:**
    • byte *substitution*: same S-box used on *every* element of the state (8 bits each)
  
  • **ShiftRows step:**
    • shift rows: *permutation* of the bytes in each row
  
  • **MixColumns step:**
    • *mix values in each column* using matrix multiplication
    • basically, applies a Hill Cipher to each column
  
  • **AddRoundkey step:**
    • XOR the state with the *round key* derived from the 128-bit encryption key
State Representation of 128-bit Block

128 bits = 16 bytes of 8 bits each – interpreted in column major order

\[(b_{0,0} | b_{1,0} | b_{2,0} | b_{3,0} | b_{0,1} | b_{1,1} | b_{2,1} | b_{3,1} | b_{0,2} | b_{1,2} | b_{2,2} | b_{3,2} | b_{0,3} | b_{1,3} | b_{2,3} | b_{3,3})\]

This array is called the State

Each group of 8 bits represented as 2 hex characters
SubBytes Step: Byte Substitution

S-boxes created:
Using number/group theory

Each byte of the state is replaced by the byte indexed by row (left 4-bits) & column (right 4-bits)
**S-Box (16 x 16)**

Example: Byte \{95\} is replaced by byte in row 9 column 5, which has value \{2A\}

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>x</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>63</td>
<td>7C</td>
<td>77</td>
<td>7B</td>
<td>F2</td>
<td>6B</td>
<td>6F</td>
<td>C5</td>
<td>30</td>
<td>01</td>
<td>67</td>
<td>2B</td>
<td>FE</td>
<td>D7</td>
<td>AB</td>
<td>76</td>
</tr>
<tr>
<td>1</td>
<td>CA</td>
<td>82</td>
<td>C9</td>
<td>7D</td>
<td>FA</td>
<td>59</td>
<td>47</td>
<td>F0</td>
<td>AD</td>
<td>D4</td>
<td>A2</td>
<td>AF</td>
<td>9C</td>
<td>A4</td>
<td>72</td>
<td>C0</td>
</tr>
<tr>
<td>2</td>
<td>B7</td>
<td>FD</td>
<td>93</td>
<td>26</td>
<td>36</td>
<td>3F</td>
<td>F7</td>
<td>CC</td>
<td>34</td>
<td>A5</td>
<td>E5</td>
<td>F1</td>
<td>71</td>
<td>D8</td>
<td>31</td>
<td>15</td>
</tr>
<tr>
<td>3</td>
<td>04</td>
<td>C7</td>
<td>23</td>
<td>C3</td>
<td>18</td>
<td>96</td>
<td>05</td>
<td>9A</td>
<td>07</td>
<td>12</td>
<td>80</td>
<td>E2</td>
<td>EB</td>
<td>27</td>
<td>B2</td>
<td>75</td>
</tr>
<tr>
<td>4</td>
<td>09</td>
<td>83</td>
<td>2C</td>
<td>1A</td>
<td>1B</td>
<td>6E</td>
<td>5A</td>
<td>A0</td>
<td>52</td>
<td>3B</td>
<td>D6</td>
<td>B3</td>
<td>29</td>
<td>E3</td>
<td>2F</td>
<td>84</td>
</tr>
<tr>
<td>5</td>
<td>53</td>
<td>D1</td>
<td>00</td>
<td>ED</td>
<td>20</td>
<td>FC</td>
<td>B1</td>
<td>5B</td>
<td>6A</td>
<td>CB</td>
<td>BE</td>
<td>39</td>
<td>4A</td>
<td>4C</td>
<td>58</td>
<td>CF</td>
</tr>
<tr>
<td>6</td>
<td>D0</td>
<td>EF</td>
<td>AA</td>
<td>FB</td>
<td>43</td>
<td>4D</td>
<td>33</td>
<td>85</td>
<td>45</td>
<td>F9</td>
<td>02</td>
<td>7F</td>
<td>50</td>
<td>3C</td>
<td>9F</td>
<td>A8</td>
</tr>
<tr>
<td>7</td>
<td>51</td>
<td>A3</td>
<td>40</td>
<td>8F</td>
<td>92</td>
<td>9D</td>
<td>38</td>
<td>F5</td>
<td>BC</td>
<td>B6</td>
<td>DA</td>
<td>21</td>
<td>10</td>
<td>FF</td>
<td>F3</td>
<td>D2</td>
</tr>
<tr>
<td>8</td>
<td>CD</td>
<td>0C</td>
<td>13</td>
<td>EC</td>
<td>5F</td>
<td>97</td>
<td>44</td>
<td>17</td>
<td>C4</td>
<td>A7</td>
<td>7E</td>
<td>3D</td>
<td>64</td>
<td>5D</td>
<td>19</td>
<td>73</td>
</tr>
<tr>
<td>9</td>
<td>60</td>
<td>81</td>
<td>4F</td>
<td>DC</td>
<td>22</td>
<td>2A</td>
<td>90</td>
<td>88</td>
<td>46</td>
<td>EE</td>
<td>B8</td>
<td>14</td>
<td>DE</td>
<td>5E</td>
<td>0B</td>
<td>DB</td>
</tr>
<tr>
<td>A</td>
<td>E0</td>
<td>32</td>
<td>3A</td>
<td>0A</td>
<td>49</td>
<td>06</td>
<td>24</td>
<td>5C</td>
<td>C2</td>
<td>D3</td>
<td>AC</td>
<td>62</td>
<td>91</td>
<td>95</td>
<td>E4</td>
<td>79</td>
</tr>
<tr>
<td>B</td>
<td>E7</td>
<td>C8</td>
<td>37</td>
<td>6D</td>
<td>BD</td>
<td>D5</td>
<td>4E</td>
<td>A9</td>
<td>6C</td>
<td>56</td>
<td>F4</td>
<td>EA</td>
<td>65</td>
<td>7A</td>
<td>AE</td>
<td>08</td>
</tr>
<tr>
<td>C</td>
<td>BA</td>
<td>78</td>
<td>25</td>
<td>2E</td>
<td>1C</td>
<td>A6</td>
<td>B4</td>
<td>C6</td>
<td>E8</td>
<td>DD</td>
<td>74</td>
<td>1F</td>
<td>4B</td>
<td>BD</td>
<td>8B</td>
<td>8A</td>
</tr>
<tr>
<td>D</td>
<td>70</td>
<td>3E</td>
<td>B5</td>
<td>66</td>
<td>48</td>
<td>03</td>
<td>F6</td>
<td>0E</td>
<td>61</td>
<td>35</td>
<td>57</td>
<td>B9</td>
<td>86</td>
<td>C1</td>
<td>1D</td>
<td>9E</td>
</tr>
<tr>
<td>E</td>
<td>E1</td>
<td>F8</td>
<td>98</td>
<td>11</td>
<td>69</td>
<td>D9</td>
<td>8E</td>
<td>94</td>
<td>9B</td>
<td>1E</td>
<td>87</td>
<td>E9</td>
<td>CE</td>
<td>55</td>
<td>28</td>
<td>DF</td>
</tr>
<tr>
<td>F</td>
<td>8C</td>
<td>A1</td>
<td>89</td>
<td>0D</td>
<td>BF</td>
<td>E6</td>
<td>42</td>
<td>68</td>
<td>41</td>
<td>99</td>
<td>2D</td>
<td>0F</td>
<td>B0</td>
<td>54</td>
<td>BB</td>
<td>16</td>
</tr>
</tbody>
</table>

*source: Table 20.2(a)*
Sub Bytes – S-Box Encrypt - Example

Input – 16 bytes of hex

| 00 | 01 | 02 | 22 |
| 23 | 24 | 25 | 80 |
| 81 | 82 | a1 | a2 |
| b3 | b4 | b5 | ff |

S-box Encrypt

Output – 16 bytes of hex

| 63 | 7c | 77 | 93 |
| 26 | 36 | 3f | cd |
| 0c | 13 | f1 | 1a |
| 4b | c6 | d2 | 16 |
**Inverse S-Box**

NOTE: This table is used for decryption

<table>
<thead>
<tr>
<th>x</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>52</td>
<td>09</td>
<td>6A</td>
<td>D5</td>
<td>30</td>
<td>36</td>
<td>A5</td>
<td>38</td>
<td>BF</td>
<td>40</td>
<td>A3</td>
<td>9E</td>
<td>81</td>
<td>F3</td>
<td>D7</td>
<td>FB</td>
</tr>
<tr>
<td>1</td>
<td>7C</td>
<td>E3</td>
<td>39</td>
<td>82</td>
<td>9B</td>
<td>2F</td>
<td>FF</td>
<td>87</td>
<td>34</td>
<td>8E</td>
<td>43</td>
<td>44</td>
<td>C4</td>
<td>DE</td>
<td>E9</td>
<td>CB</td>
</tr>
<tr>
<td>2</td>
<td>54</td>
<td>7B</td>
<td>94</td>
<td>32</td>
<td>A6</td>
<td>C2</td>
<td>23</td>
<td>3D</td>
<td>EE</td>
<td>4C</td>
<td>95</td>
<td>0B</td>
<td>42</td>
<td>FA</td>
<td>C3</td>
<td>4E</td>
</tr>
<tr>
<td>3</td>
<td>08</td>
<td>2E</td>
<td>A1</td>
<td>66</td>
<td>28</td>
<td>D9</td>
<td>24</td>
<td>B2</td>
<td>76</td>
<td>5B</td>
<td>A2</td>
<td>49</td>
<td>6D</td>
<td>8B</td>
<td>D1</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>72</td>
<td>F8</td>
<td>F6</td>
<td>64</td>
<td>86</td>
<td>68</td>
<td>98</td>
<td>16</td>
<td>D4</td>
<td>A4</td>
<td>5C</td>
<td>CC</td>
<td>5D</td>
<td>65</td>
<td>B6</td>
<td>92</td>
</tr>
<tr>
<td>5</td>
<td>6C</td>
<td>70</td>
<td>48</td>
<td>50</td>
<td>FD</td>
<td>ED</td>
<td>B9</td>
<td>DA</td>
<td>5E</td>
<td>15</td>
<td>46</td>
<td>57</td>
<td>A7</td>
<td>8D</td>
<td>9D</td>
<td>84</td>
</tr>
<tr>
<td>6</td>
<td>90</td>
<td>D8</td>
<td>AB</td>
<td>00</td>
<td>8C</td>
<td>BC</td>
<td>D3</td>
<td>0A</td>
<td>F7</td>
<td>E4</td>
<td>58</td>
<td>05</td>
<td>B8</td>
<td>B3</td>
<td>45</td>
<td>06</td>
</tr>
<tr>
<td>7</td>
<td>D0</td>
<td>2C</td>
<td>1E</td>
<td>8F</td>
<td>CA</td>
<td>3F</td>
<td>0F</td>
<td>02</td>
<td>C1</td>
<td>AF</td>
<td>BD</td>
<td>03</td>
<td>13</td>
<td>8A</td>
<td>6B</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3A</td>
<td>91</td>
<td>11</td>
<td>41</td>
<td>4F</td>
<td>67</td>
<td>DC</td>
<td>EA</td>
<td>97</td>
<td>F2</td>
<td>CF</td>
<td>CE</td>
<td>F0</td>
<td>B4</td>
<td>E6</td>
<td>73</td>
</tr>
<tr>
<td>9</td>
<td>96</td>
<td>AC</td>
<td>74</td>
<td>22</td>
<td>E7</td>
<td>AD</td>
<td>35</td>
<td>85</td>
<td>E2</td>
<td>F9</td>
<td>37</td>
<td>E8</td>
<td>1C</td>
<td>75</td>
<td>DF</td>
<td>6E</td>
</tr>
<tr>
<td>A</td>
<td>47</td>
<td>F1</td>
<td>1A</td>
<td>71</td>
<td>1D</td>
<td>29</td>
<td>C5</td>
<td>89</td>
<td>6F</td>
<td>B7</td>
<td>62</td>
<td>0E</td>
<td>AA</td>
<td>18</td>
<td>BE</td>
<td>1B</td>
</tr>
<tr>
<td>B</td>
<td>FC</td>
<td>56</td>
<td>3E</td>
<td>4B</td>
<td>C6</td>
<td>D2</td>
<td>79</td>
<td>20</td>
<td>9A</td>
<td>DB</td>
<td>C0</td>
<td>FE</td>
<td>78</td>
<td>CD</td>
<td>5A</td>
<td>F4</td>
</tr>
<tr>
<td>C</td>
<td>1F</td>
<td>DD</td>
<td>A8</td>
<td>33</td>
<td>88</td>
<td>07</td>
<td>C7</td>
<td>31</td>
<td>B1</td>
<td>12</td>
<td>10</td>
<td>59</td>
<td>27</td>
<td>80</td>
<td>EC</td>
<td>5F</td>
</tr>
<tr>
<td>D</td>
<td>60</td>
<td>51</td>
<td>7F</td>
<td>A9</td>
<td>19</td>
<td>B5</td>
<td>4A</td>
<td>0D</td>
<td>2D</td>
<td>E5</td>
<td>7A</td>
<td>9F</td>
<td>93</td>
<td>C9</td>
<td>9C</td>
<td>EF</td>
</tr>
<tr>
<td>E</td>
<td>A0</td>
<td>E0</td>
<td>3B</td>
<td>4D</td>
<td>AE</td>
<td>2A</td>
<td>F5</td>
<td>B0</td>
<td>C8</td>
<td>EB</td>
<td>BB</td>
<td>3C</td>
<td>83</td>
<td>53</td>
<td>99</td>
<td>61</td>
</tr>
<tr>
<td>F</td>
<td>17</td>
<td>2B</td>
<td>04</td>
<td>7E</td>
<td>BA</td>
<td>77</td>
<td>D6</td>
<td>26</td>
<td>E1</td>
<td>69</td>
<td>14</td>
<td>63</td>
<td>55</td>
<td>21</td>
<td>0C</td>
<td>7D</td>
</tr>
</tbody>
</table>

*source: Table 20.2(b)*
### ShiftRows Step

<table>
<thead>
<tr>
<th>First Row</th>
<th>Second Row</th>
<th>Third Row</th>
<th>Fourth Row</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_{0,0}$</td>
<td>$s_{0,1}$</td>
<td>$s_{0,2}$</td>
<td>$s_{0,3}$</td>
</tr>
<tr>
<td>$s_{1,0}$</td>
<td>$s_{1,1}$</td>
<td>$s_{1,2}$</td>
<td>$s_{1,3}$</td>
</tr>
<tr>
<td>$s_{2,0}$</td>
<td>$s_{2,1}$</td>
<td>$s_{2,2}$</td>
<td>$s_{2,3}$</td>
</tr>
<tr>
<td>$s_{3,0}$</td>
<td>$s_{3,1}$</td>
<td>$s_{3,2}$</td>
<td>$s_{3,3}$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>First Row</th>
<th>Second Row</th>
<th>Third Row</th>
<th>Fourth Row</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s'_{0,0}$</td>
<td>$s'_{0,1}$</td>
<td>$s'_{0,2}$</td>
<td>$s'_{0,3}$</td>
</tr>
<tr>
<td>$s'_{1,0}$</td>
<td>$s'_{1,1}$</td>
<td>$s'_{1,2}$</td>
<td>$s'_{1,3}$</td>
</tr>
<tr>
<td>$s'_{2,0}$</td>
<td>$s'_{2,1}$</td>
<td>$s'_{2,2}$</td>
<td>$s'_{2,3}$</td>
</tr>
<tr>
<td>$s'_{3,0}$</td>
<td>$s'_{3,1}$</td>
<td>$s'_{3,2}$</td>
<td>$s'_{3,3}$</td>
</tr>
</tbody>
</table>

1\textsuperscript{st} row is unchanged  
2\textsuperscript{nd} row does 1 byte circular shift to left  
3\textsuperscript{rd} row does 2 byte circular shift to left  
4\textsuperscript{th} row does 3 byte circular shift to left  

**NOTE:** same set of shifts every time
MixColumns Step

- Each column is processed separately
- Each byte is replaced by a value dependent on all 4 bytes in the column
- Mix columns matrix is part of AES, just like the S-box and Inverse S-box

**Mix Columns Matrix**

<table>
<thead>
<tr>
<th></th>
<th>State</th>
<th>New State</th>
</tr>
</thead>
</table>
| $\begin{bmatrix}
02 & 03 & 01 & 01 \\
01 & 02 & 03 & 01 \\
01 & 01 & 02 & 03 \\
03 & 01 & 01 & 02 \\
\end{bmatrix}$ | $\begin{bmatrix}
S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\
S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\
S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\
S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \\
\end{bmatrix}$ | $\begin{bmatrix}
S_{0,0}' & S_{0,1}' & S_{0,2}' & S_{0,3}' \\
S_{1,0}' & S_{1,1}' & S_{1,2}' & S_{1,3}' \\
S_{2,0}' & S_{2,1}' & S_{2,2}' & S_{2,3}' \\
S_{3,0}' & S_{3,1}' & S_{3,2}' & S_{3,3}' \\
\end{bmatrix}$ |

**Example:** $s_{1,2}' = 1 \cdot s_{0,2} + 2 \cdot s_{1,2} + 3 \cdot s_{2,2} + 1 \cdot s_{3,2}$
Mix columns fun... with Galois

- AES security is entirely based on GF(2^8) (see Galois Fields) using irreducible polynomial:
  - \( x^8 + x^4 + x^3 + x + 1 \ldots \)
- A byte is not interpreted as binary but
  - As a special polynomial of at most degree 7
- \( 63_x \rightarrow 0110\ 0011 \)
  - \( x^6 + x^5 + x + 1 \)
- \( 63_x = 99_{10} \)
  - \( 2^6 + 2^5 + 2 + 1 \)
  - Replace 2 with \( x \) to make the polynomial
- GF(2^8)
  - 8 coefficients
  - Each with value of \{0,1\}
- Big idea?
  - Define addition, subtraction & multiplication on the polynomials
    - Have closure
    - Be valid polynomials
Wait? There’s a problem?

- Calculate coefficients Mod 2 (similar to XOR)
  - \((x^6 + x^5 + x + 1) + (x^3 + x^2 + x)\)
    \[ = (x^6 + x^5 + x^3 + x^2 + 2x + 1) \]
    \[ = \text{REDUCE coefficients Mod 2 (Xor)} \]
    \[ = (x^6 + x^5 + x^3 + x^2 + 1) \]
- Note that subtraction is identical to addition & both are essentially like Xor
- Hard part?
- Multiplication
  - \((x + 1)(x^6 + x^5 + x + 1)\)
    \[ = (x^7 + x^6 + x^2 + x) + (x^6 + x^5 + x + 1) \]
    \[ = x^7 + 2x^6 + x^5 + x^2 + 2x + 1 \]
    \[ = x^7 + x^5 + x^2 + 1 \]
Hmmm... About that problem

- Multiplying by \( x \) is usually left shifting the bits, in this case 8 of ‘em, 1 bit to the left then Xor the shifted bits with the original value
  
  \[
  \begin{align*}
  0110 0011 \\
  1100 0110 \\
  = 1010 0101
  \end{align*}
  \]

- The problem?
  
  \[
  x(x^7 + x^5 + x^2 + 1)
  
  x^8 + x^6 + x^3 + x
  \]

- ... overflow

- So, mod by the AES (GFx^8) factor

- \( x^8 + x^6 + x^3 + x \mod x^8 + x^4 + x^3 + x +1 \)

- Consider \( x^8 \Rightarrow x^4 + x^3 + x + 1 \)

- THEREFORE
  
  - See \( x^8 \)?
  - Replace with \( x^4 + x^3 + x + 1 \)
The solution – in example\(^1\)

- Recap, given
  \[x^8 + x^6 + x^3 + x\]

- Replace \(x^8\) with \(x^4 + x^3 + x + 1\) then
  \[(x^4 + x^3 + x + 1) + x^6 + x^3 + x\]
  \[x^6 + x^4 + 1\]

- Consider
  
  \[
  \begin{array}{c}
  1010 0101 \text{ multiply by 2} \\
  1 0100 1010 \\
  0100 1010 \\
  0001 1011 \\
  0101 0001
  \end{array}
  \]
  
- Same algorithm as shown earlier
The solution – in example²

• Given

• \((x + 1)(x^7 + x^5 + x^2 + 1)\)
  \[x(x^7 + x^5 + x^2 + 1) + (x^7 + x^5 + x^2 + 1)\]
  \[x^6 + x^4 + 1 + (x^7 + x^5 + x^2 + 1)\]
  \[x^7 + x^6 + x^5 + x^4 + x^2\]

• Hmmm... 03 x A5 = F4

  \((x + 1)(x^7 + x^5 + x^2 + 1) = x^7 + x^6 + x^5 + x^4 + x^2\)
Mixin’ some columns...

- Given the **FIXED** matrix

<table>
<thead>
<tr>
<th>02</th>
<th>03</th>
<th>01</th>
<th>01</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>02</td>
<td>03</td>
<td>01</td>
</tr>
<tr>
<td>01</td>
<td>01</td>
<td>02</td>
<td>03</td>
</tr>
<tr>
<td>03</td>
<td>01</td>
<td>01</td>
<td>02</td>
</tr>
</tbody>
</table>

\[
\begin{array}{cccc}
63 & 7c & 77 & 93 \\
36 & 3f & cd & 26 \\
f1 & 1a & 0c & 13 \\
16 & 4b & c6 & d2 \\
\end{array}
\]

\[02 \times 7c + 03 \times 3f + 01 \times 1a + 01 \times 4b = e8\]
AddRoundKey Step

Exclusive-or (XOR) the state with a set of keys for the round, derived from the 128-bit secret key.
The 128-bit key is first divided into four 4-byte "words" (represented as columns in diagram)

First column of round key is computed as XOR of previous round first column and the “T” transformation of the previous round last column.

“T” transformation involves
- Cyclical left shifts
- S-box substitutions
- XOR first byte with a “round” constant

Remaining columns computed in order using XOR of previous round column value and the preceding column in the current round.
AES Encryption Round

source: Fig. 20.4
### AES Assembler instructions

- **For Intel & AMD:**

<table>
<thead>
<tr>
<th>Instruction</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AESENC</td>
<td>Perform one round of an AES encryption flow</td>
</tr>
<tr>
<td>AESENCLAST</td>
<td>Perform the last round of an AES encryption flow</td>
</tr>
<tr>
<td>AESDEC</td>
<td>Perform one round of an AES decryption flow</td>
</tr>
<tr>
<td>AESDECLAST</td>
<td>Perform the last round of an AES decryption flow</td>
</tr>
<tr>
<td>AESKEYGENASSIST</td>
<td>Assist in AES round key generation</td>
</tr>
<tr>
<td>AESIMC</td>
<td>Assist in AES Inverse Mix Columns</td>
</tr>
<tr>
<td>PCLMULQDQ</td>
<td>Carryless multiply (CLMUL)</td>
</tr>
</tbody>
</table>
Resources

• NIST Specification
  • https://www.nist.gov/publications/advanced-encryption-standard-aes

• Intel – AES instructions w. explanation

• Source code library (in C)
  • https://github.com/BrianGladman/aes