Title: A simple proof of the threshold for fault-tolerant quantum computation

Date: Mar 23, 2005 04:00 PM

URL: http://pirsa.org/05030120

Abstract: One of the central critical results in the theory of fault-tolerant quantum computation is that arbitrarily long reliable computation is possible provided the error rate per gate and per time step is below some threshold value. This was proved by a number of groups, but the detailed published proofs are complex and furthermore only hold for concatenation of quantum error-correcting codes able to correct 2 errors per block, while typically the best estimates of the threshold value are based on the 7-qubit code, which only corrects 1 error per block. I will describe recent work by Panos Aliferis, John Preskill, and myself which substantially simplifies existing proofs and applies as well to the concatenated 7-qubit code. The new proof also provides a nice framework in which to attempt to prove relatively high values of the threshold, which so far have only emerged as estimates from simulations

A Simple Proof of the Threshold for Fault-Tolerance

Panos Aliferis

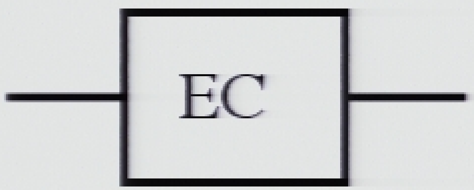
Daniel Gottesman

John Preskill

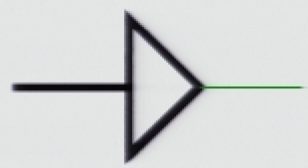
Fault-Tolerance

Quantum states can be encoded in quantum error-correcting codes (QECCs) to protect against errors, but we need fault-tolerant protocols to perform computations on encoded states.

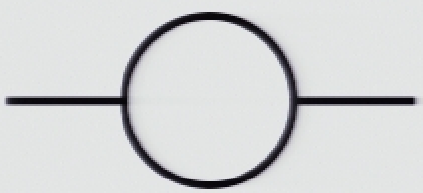
We need:



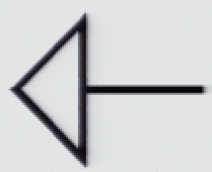
Fault-tolerant error correction



Fault-tolerant measurement



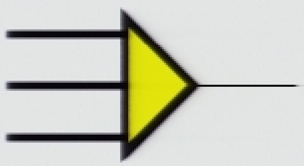
Fault-tolerant encoded gates



Fault-tolerant preparation of encoded state.

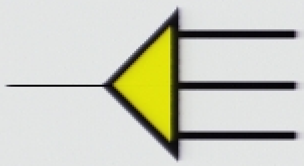
Ideal Decoders and Encoder

If a block of a QECC has errors, how do we define the state of the encoded data?



Ideal decoder

Corrects errors and decodes state producing an unencoded qubit.



Ideal encoder

Produces a perfect encoded state from an unencoded qubit.

These operations cannot be performed using real gates, but they are useful for defining and proving fault-tolerance. For instance, they preserve entanglement correctly.

Some Conventions

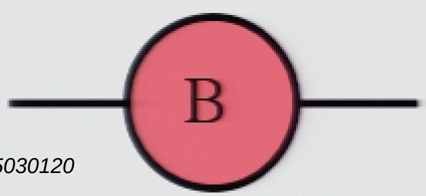
Unencoded qubit (thin black)

Classical bit (thin green)

Encoded qubit (thick black; other colors for concatenation)

VG = “very good” = no errors (yellow fill)

G = “good” = 0 or 1 errors (no fill)



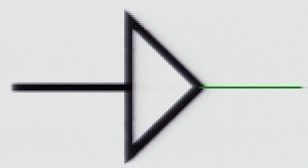
B = “bad” = 2 or more errors (red fill)

Fault-Tolerance

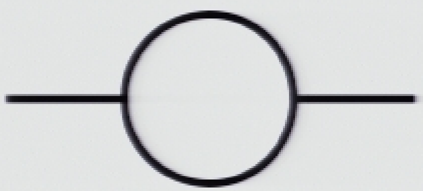
Quantum states can be encoded in quantum error-correcting codes (QECCs) to protect against errors, but we need fault-tolerant protocols to perform computations on encoded states.

We need:

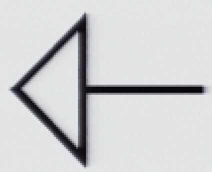
Fault-tolerant error correction



Fault-tolerant measurement



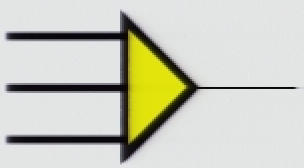
Fault-tolerant encoded gates



Fault-tolerant preparation of encoded state.

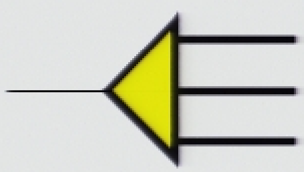
Ideal Decoders and Encoder

If a block of a QECC has errors, how do we define the state of the encoded data?



Ideal decoder

Corrects errors and decodes state producing an unencoded qubit.



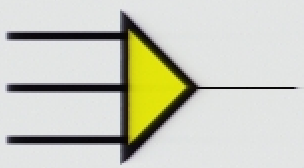
Ideal encoder

Produces a perfect encoded state from an unencoded qubit.

These operations cannot be performed using real gates, but they are useful for defining and proving fault-tolerance. For instance, they preserve entanglement correctly.

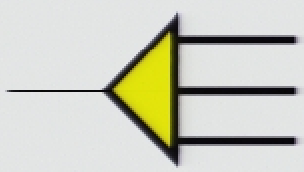
Ideal Decoders and Encoder

If a block of a QECC has errors, how do we define the state of the encoded data?



Ideal decoder

Corrects errors and decodes state producing an unencoded qubit.



Ideal encoder

Produces a perfect encoded state from an unencoded qubit.

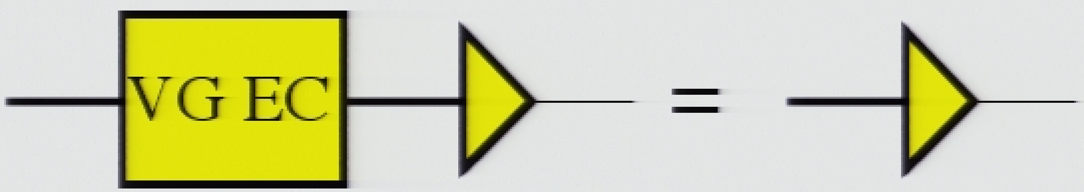
These operations cannot be performed using real gates, but they are useful for defining and proving fault-tolerance. For instance, they preserve entanglement correctly.

Properties of FT Error Correction

1.

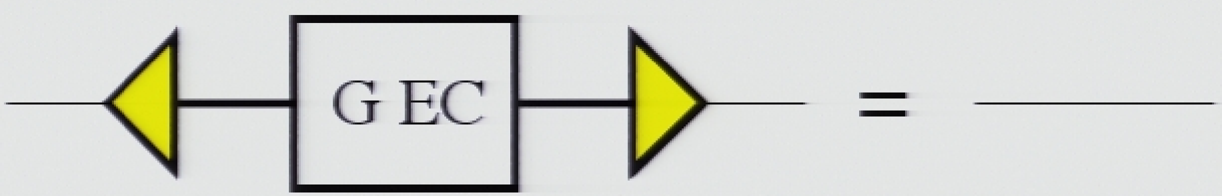
A very good error correction leaves a state with no errors.

2.



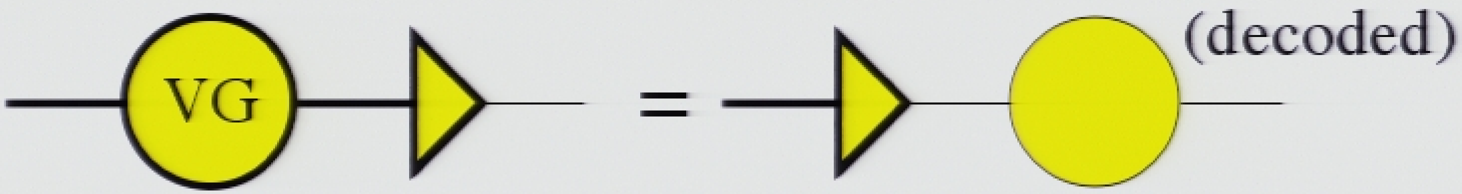
An ideal decoder includes very good error correction.

3.

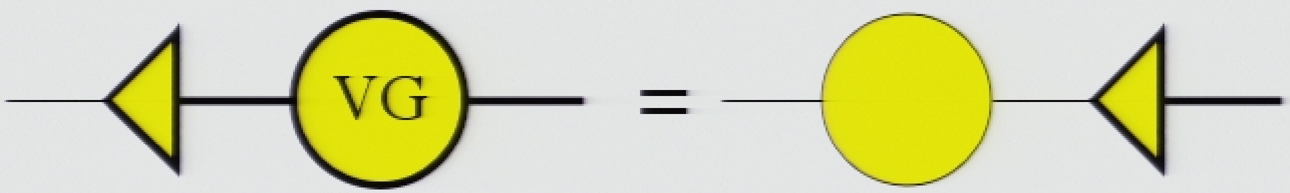


A good error correction acting on a state with no errors leaves a state that has at most 1 error (i.e., it can be correctly decoded).

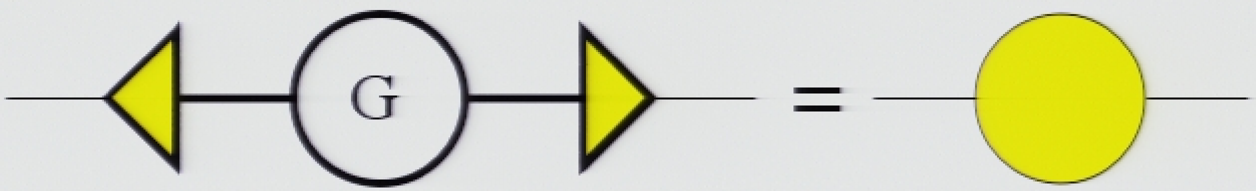
Properties of FT gates

4. 

A very good gate acts on any encoded state just like a perfect gate on the decoding of the state.

5. 

A very good gate does not create errors. (A good gate would not work, since it could leave an error behind.)

6. 

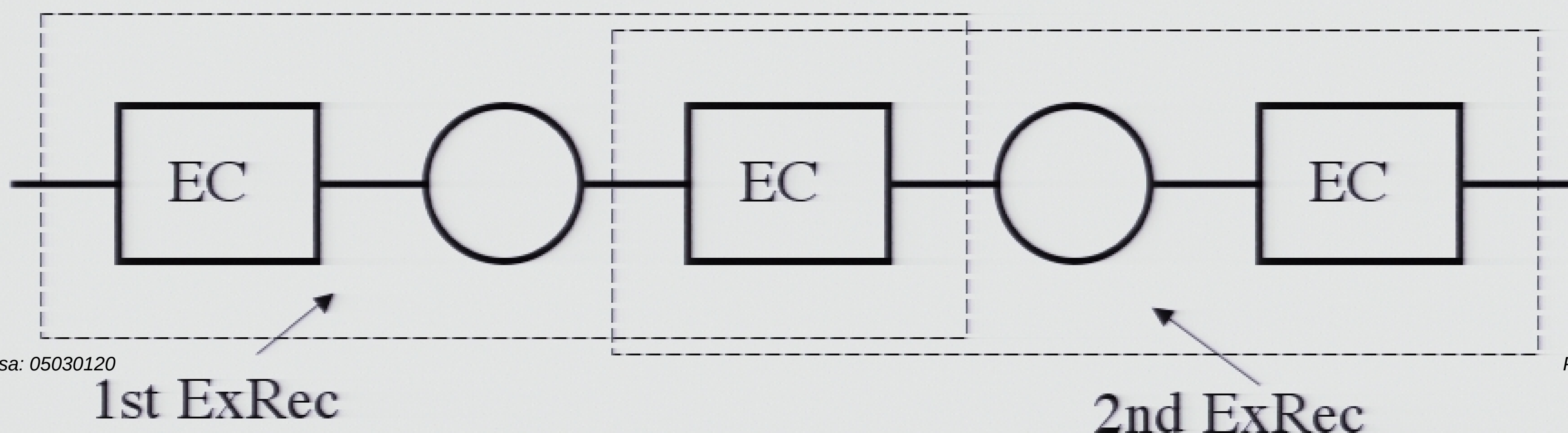
A good gate acting on a state with no errors acts the same as a perfect gate on an uncoded state.

Extended Rectangles

Definition: An "extended rectangle" (or "ExRec") consists of an EC step ("leading"), followed by an encoded gate, followed by another EC step ("trailing").

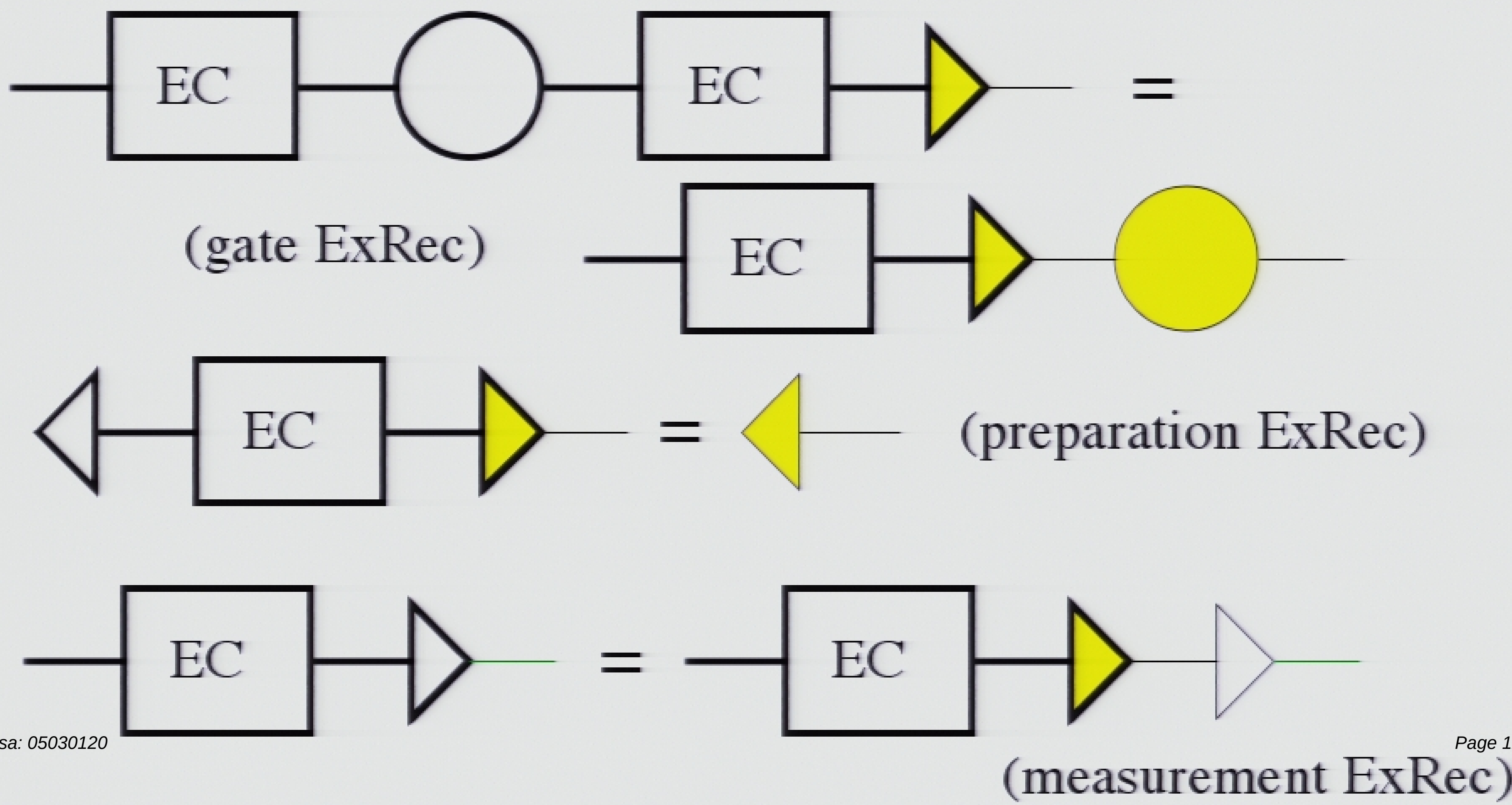
Definition: An ExRec is "good" if it contains at most one fault (a physical gate or time step with an error). Otherwise it is "bad".

Note: Extended rectangles overlap with each other.



Good Circuits are Correct

Lemma [ExRec-Cor]: An ideal decoder can be pulled back through a good ExRec to just after the leading EC.



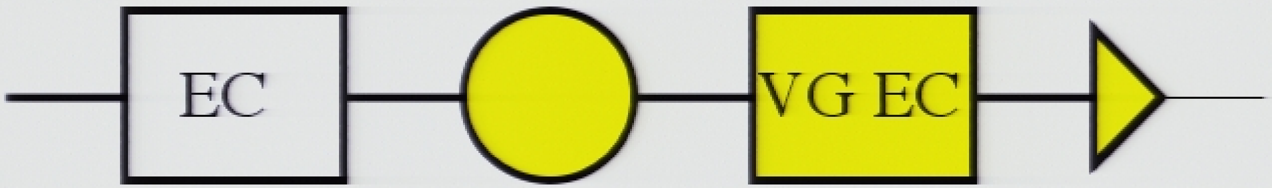
Good Circuits are Correct (Proof)

The ExRec is good, so there is at most one error.

Good Circuits are Correct (Proof)

The ExRec is good, so there is at most one error.

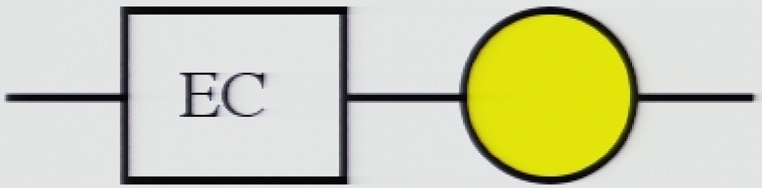
Case I: There is no error or one in the leading EC.



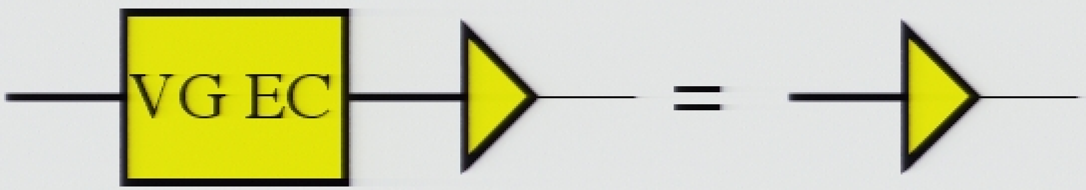
Good Circuits are Correct (Proof)

The ExRec is good, so there is at most one error.

Case I: There is no error or one in the leading EC.



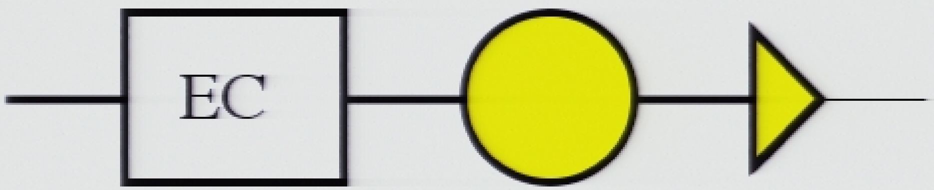
2.

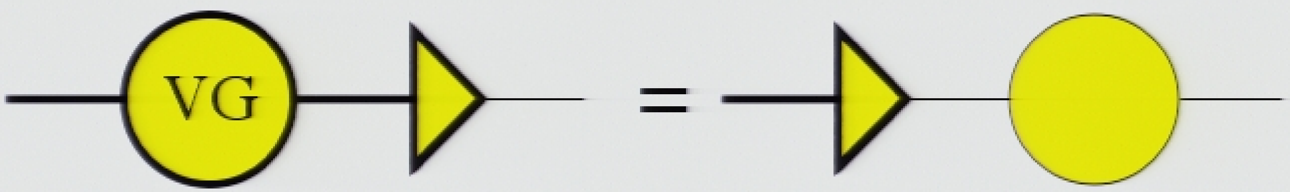


Good Circuits are Correct (Proof)

The ExRec is good, so there is at most one error.

Case I: There is no error or one in the leading EC.

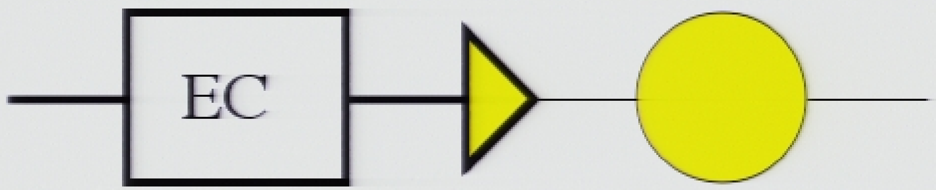


4. 

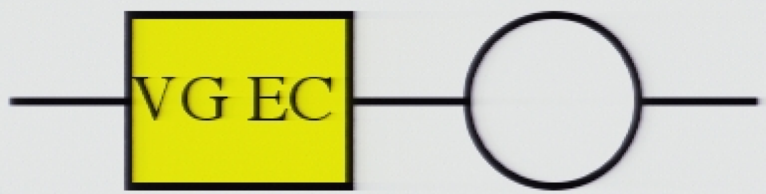
Good Circuits are Correct (Proof)

The ExRec is good, so there is at most one error.

Case I: There is no error or one in the leading EC.



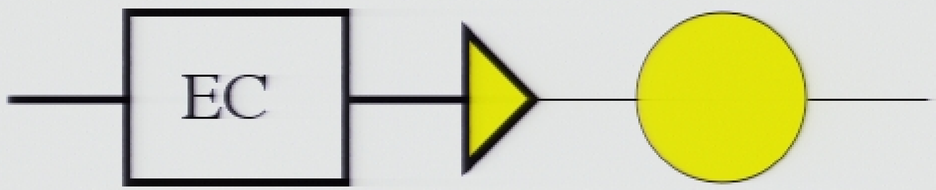
Case II: There is an error in the gate.



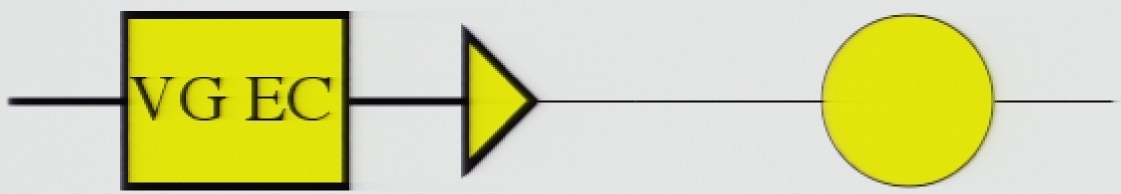
Good Circuits are Correct (Proof)

The ExRec is good, so there is at most one error.

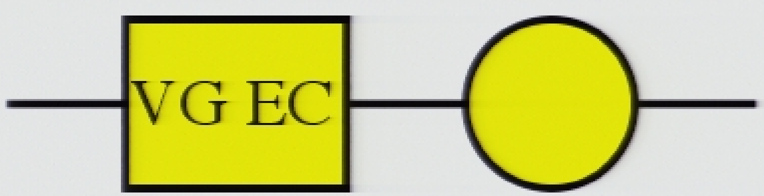
Case I: There is no error or one in the leading EC.



Case II: There is an error in the gate.



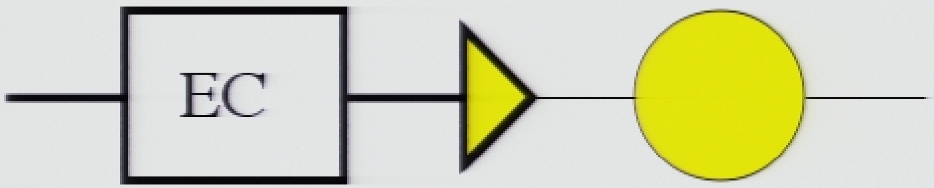
Case III: There is an error in the trailing EC.



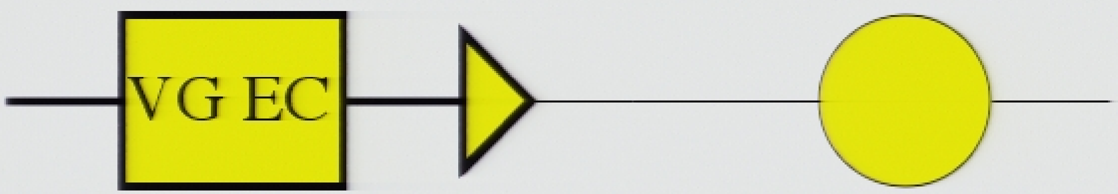
Good Circuits are Correct (Proof)

The ExRec is good, so there is at most one error.

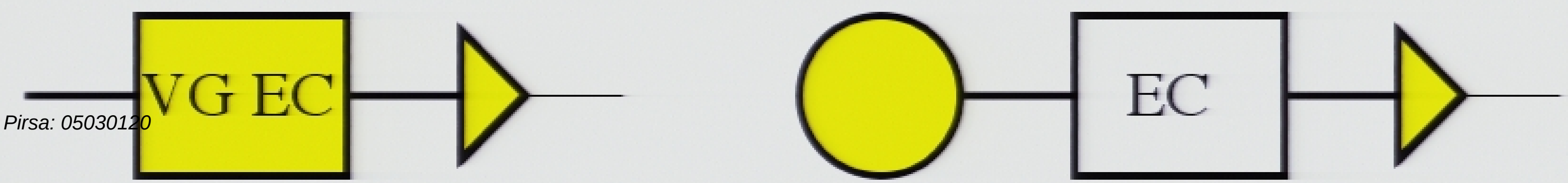
Case I: There is no error or one in the leading EC.



Case II: There is an error in the gate.

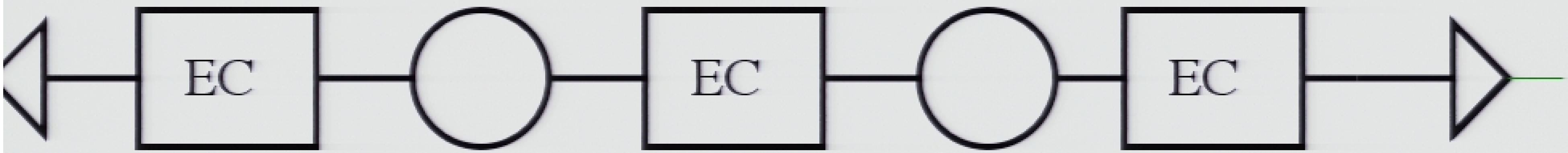


Case III: There is an error in the trailing EC.



Correct Means What It Ought To

Suppose we have a circuit consisting of only good ExRecs. Then its action is equivalent to that of the corresponding ideal circuit:



Correct Means What It Ought To

Suppose we have a circuit consisting of only good ExRecs. Then its action is equivalent to that of the corresponding ideal circuit:

1. Use ExRec-Cor for measurement to introduce an ideal decoder before the final measurement.
2. Use ExRec-Cor for gates to push the ideal decoder back to just after the very first EC step.

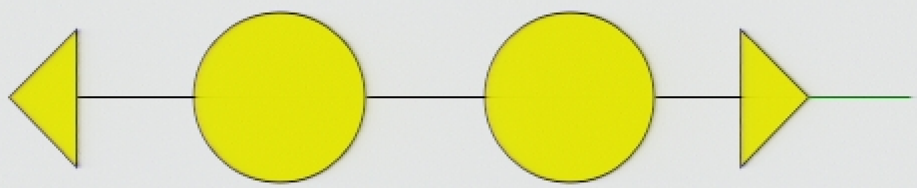
Correct Means What It Ought To

Suppose we have a circuit consisting of only good ExRecs. Then its action is equivalent to that of the corresponding ideal circuit:

1. Use ExRec-Cor for measurement to introduce an ideal decoder before the final measurement.
2. Use ExRec-Cor for gates to push the ideal decoder back to just after the very first EC step.

Correct Means What It Ought To

Suppose we have a circuit consisting of only good ExRecs. Then its action is equivalent to that of the corresponding ideal circuit:



1. Use ExRec-Cor for measurement to introduce an ideal decoder before the final measurement.
2. Use ExRec-Cor for gates to push the ideal decoder back to just after the very first EC step.
3. Use ExRec-Cor for preparation to eliminate the decoder.

Fault-Tolerance Reduces Errors

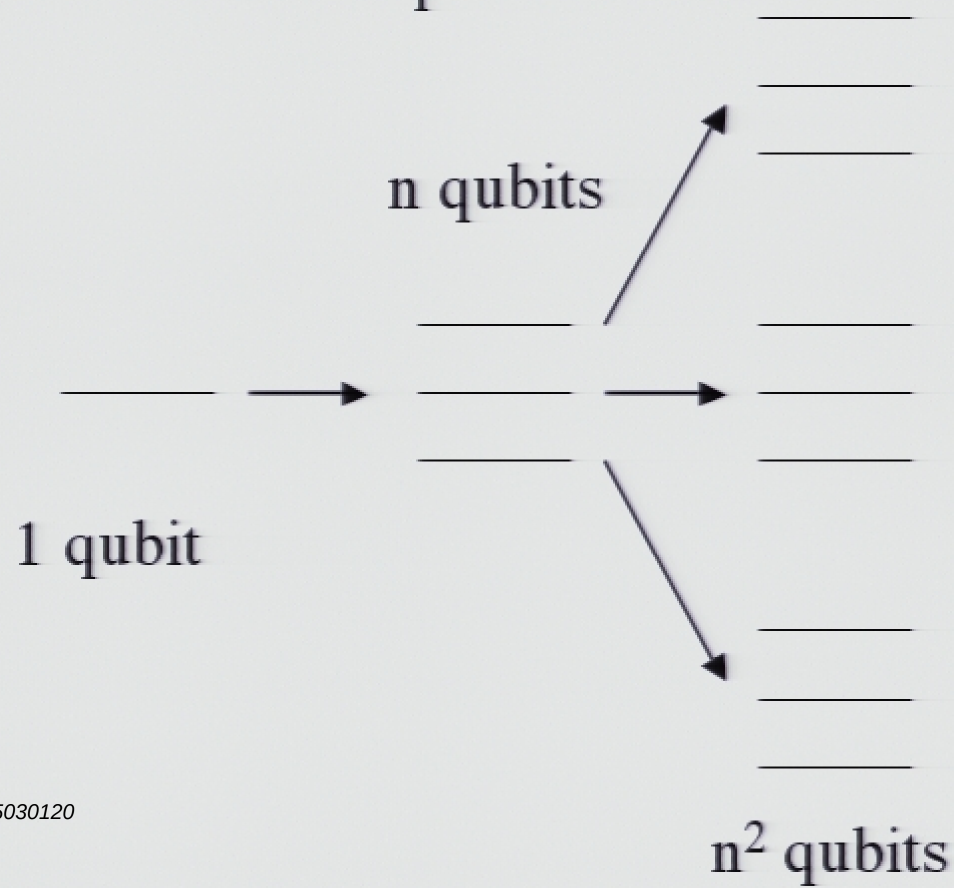
Theorem: A calculation with T locations (gates & idle time steps) and an error rate p per physical location can be encoded in a fault-tolerant computation with an overall failure rate at most CTp^2, for some constant C.

Note: An unencoded computation would fail with prob. Tp.

Proof: If all ExRecs in the fault-tolerant computation are good, the computation gives the right answer. The probability of a single ExRec being bad is $C_R\, p^2$, where C_R is the number of pairs of locations in the ExRec. If C is the maximum value of C_R for all ExRecs, the total probability of at least one ExRec being bad is at most CTp^2.

Threshold from Concatenation

Suppose we have a QECC encoding 1 qubit as n, and correcting t errors. Encode each physical qubit using the same QECC for even more error protection.



Threshold: If the error rate per gate and time step is less than some threshold p_t, then arbitrarily long reliable quantum computations are possible.

Concatenation and ExRecs

To add a level of concatenation, replace each physical gate in the fault-tolerant circuit with a new copy of the fault-tolerant circuit for that gate, and intersperse new EC steps between gates.

Level 1 blocks contain n qubits, and level k blocks contain n^k.

Definition: A gate ExRec at level k consists of a level-k EC step ("leading"), followed by a level-k gate, followed by another level-k EC step ("trailing"), and similarly for preparation and measurement ExRecs.

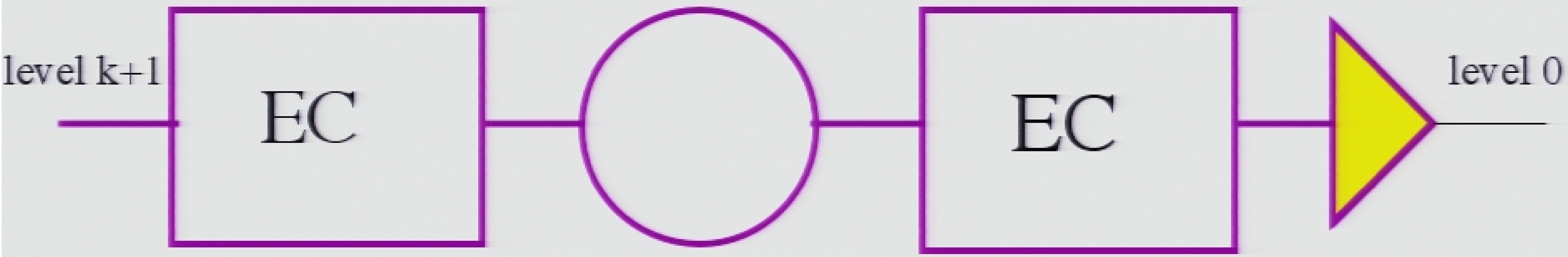
Definition: An ExRec at level k is good if at most one of its constituent level k-1 sub-ExRecs is bad.*

* Not exactly, but we will correct it later.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

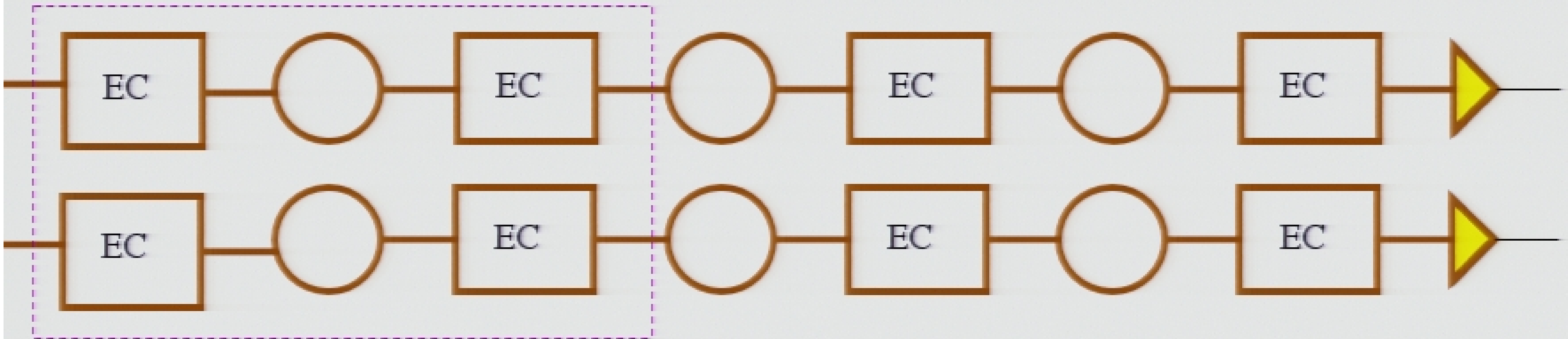
Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.



Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

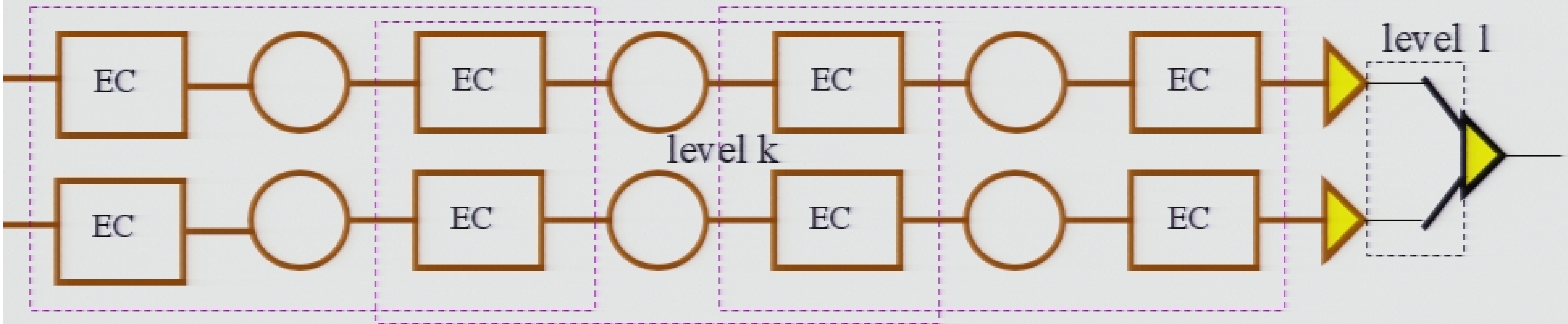


1. Interpret level k+1 ExRec as a series of level k gates and level k+1 decoded as level k decoders followed by level 1 decoder.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

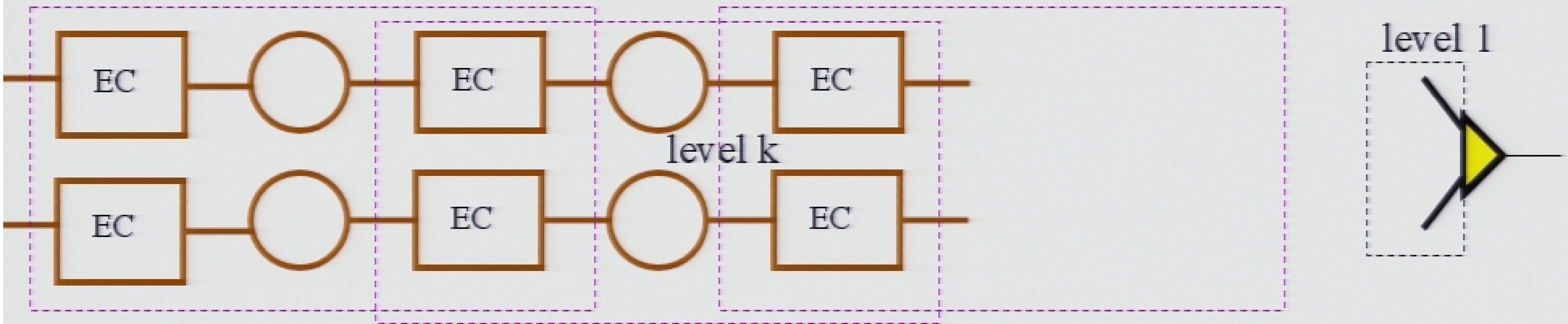


1. Interpret k+1 as k-ExRecs.
2. Push back the level k decoders to just after the first level k ECs.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

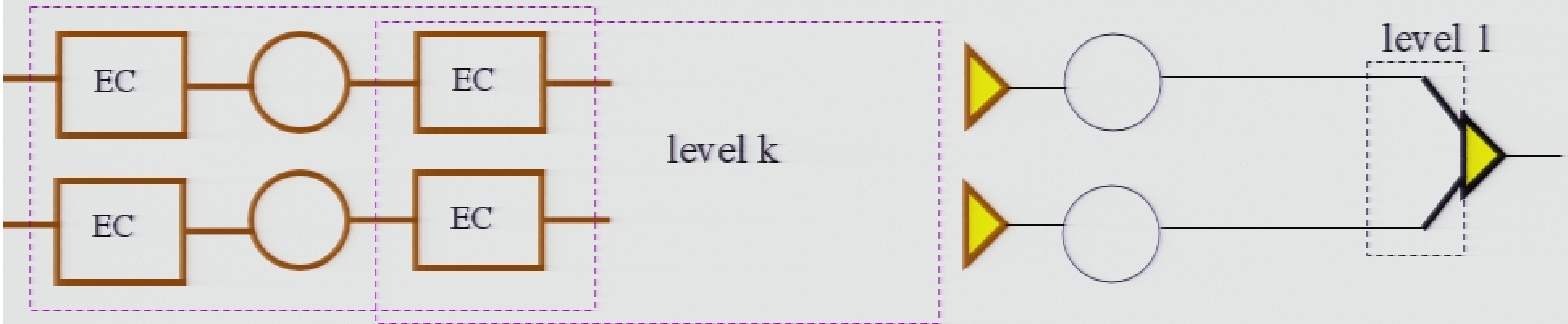


1. Interpret k+1 as k-ExRecs.
2. Push back the level k decoders to just after the first level k ECs.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

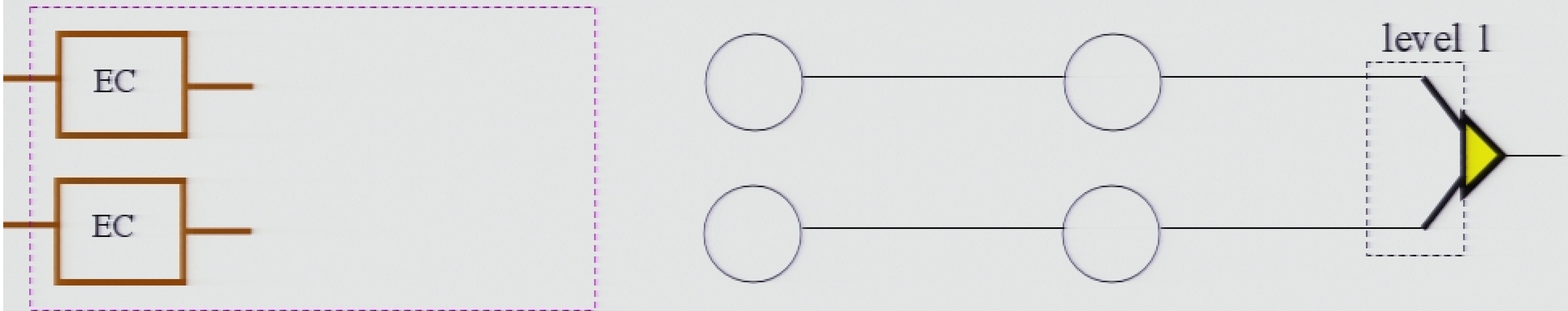


1. Interpret k+1 as k-ExRecs.
2. Push back the level k decoders to just after the first level k ECs.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

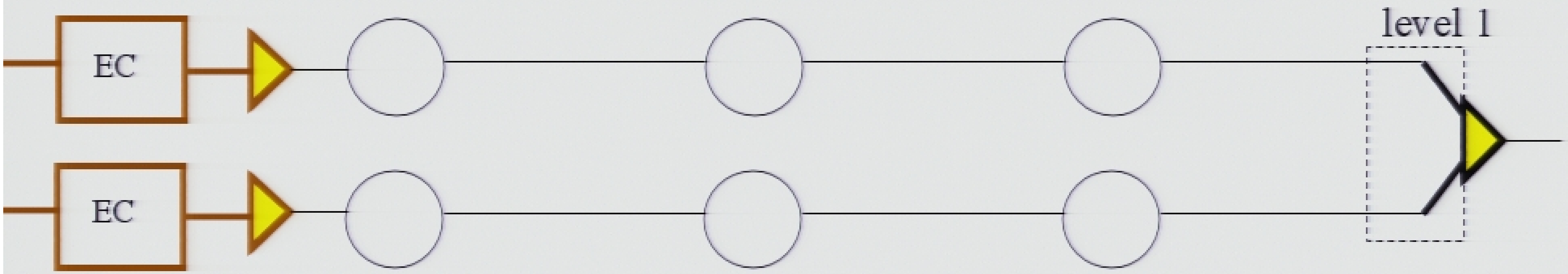


1. Interpret k+1 as k-ExRecs.
2. Push back the level k decoders to just after the first level k ECs.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.



1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

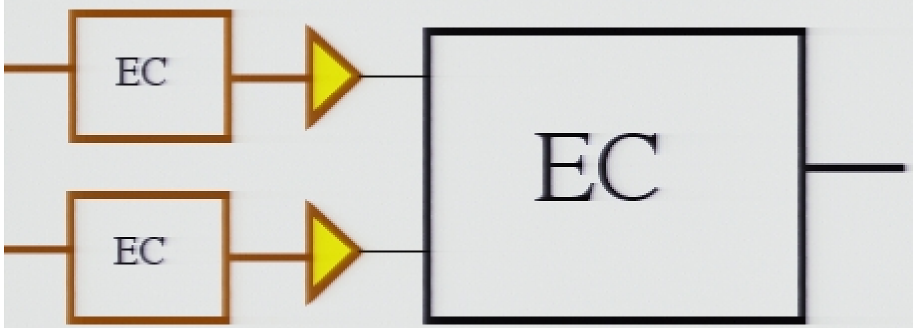
Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.
4. Push back level 1 decoder to after first level 1 EC.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.



1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.
4. Push back level 1 decoder to after first level 1 EC.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

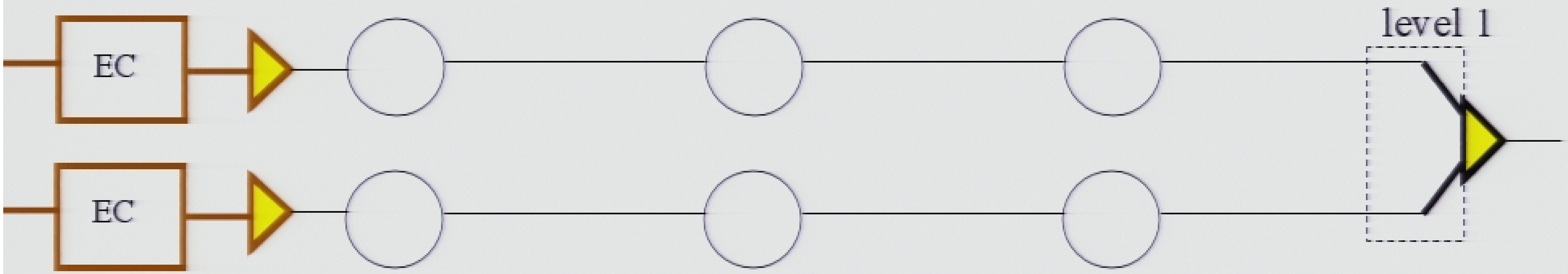
Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.
4. Push back level 1 decoder to after first level 1 EC.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.



1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

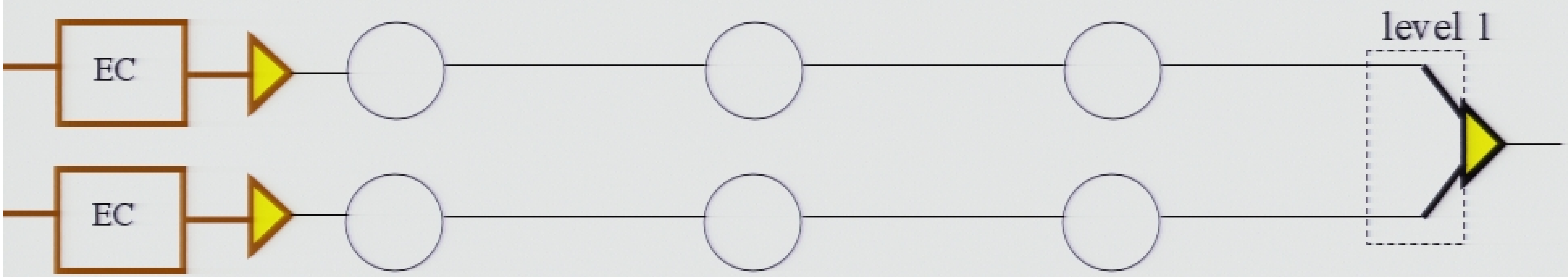
Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.



1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

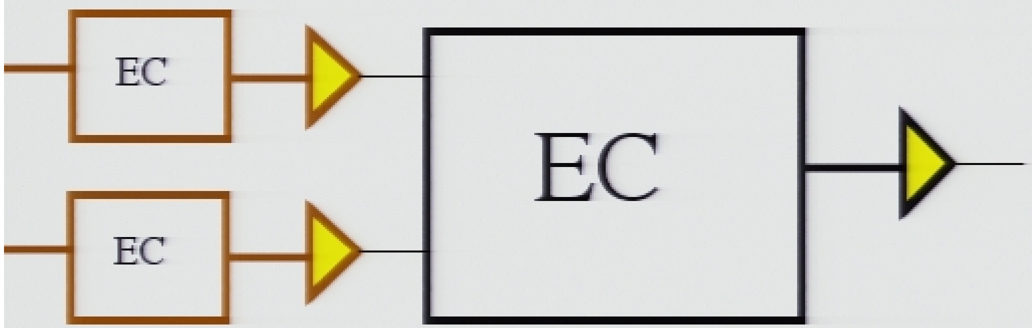
Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.



1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.
4. Push back level 1 decoder to after first level 1 EC.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

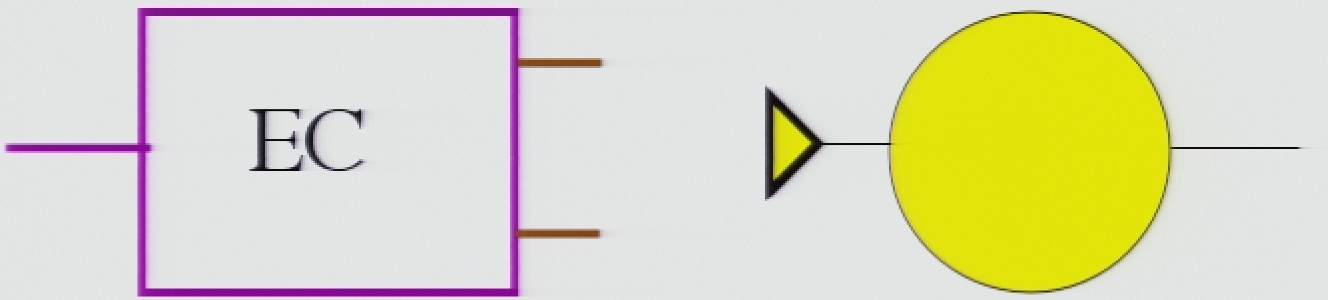
Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.

1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.
4. Push back level 1 decoder.
5. Push level k decoders forward to after level 1 EC.

Good Circuits Are Correct (Reprise)

Lemma: If ExRec-Cor holds at level 1, it holds at level k.

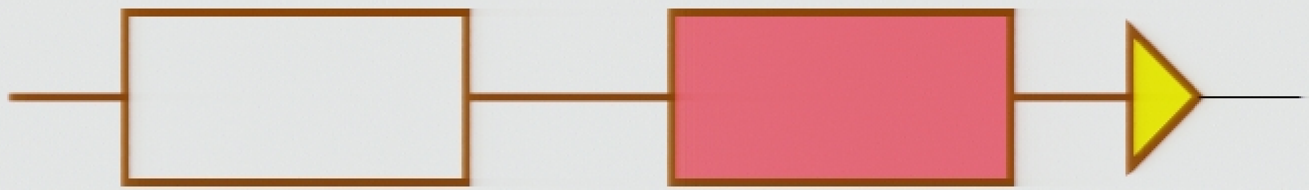
Proof: Suppose it holds at level k. We will prove it at level k+1. First consider the case when there are no bad sub-ExRecs.



1. Interpret k+1 as k-ExRecs.
2. Push k-decoders back to beginning of circuit.
3. Interpret level 0 gates as a level 1 ExRec.
4. Push back level 1 decoder.
5. Push level k decoders forward to after level 1 EC.

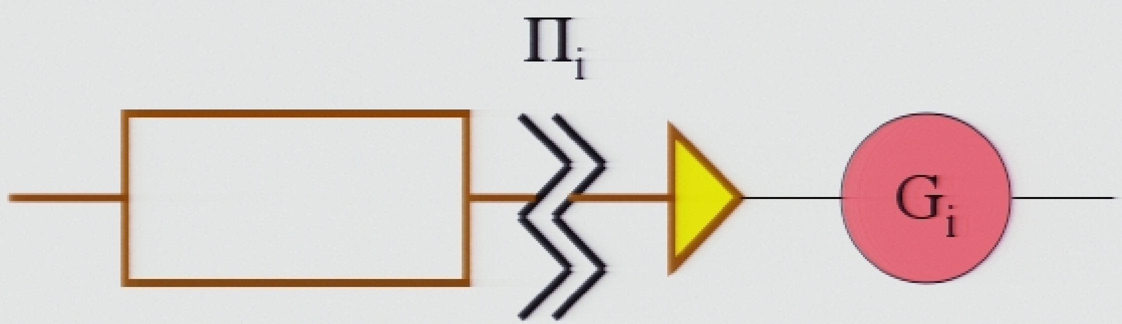
Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.



Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.

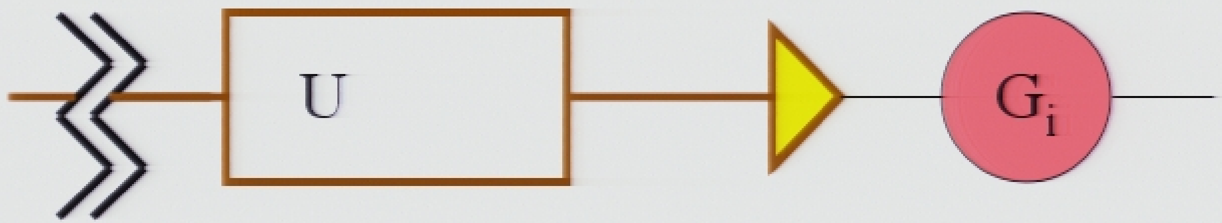


1. Insert rank 1 projectors $\sum \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.

Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.

$\Pi_i' = U\Pi_i U^{-1}$

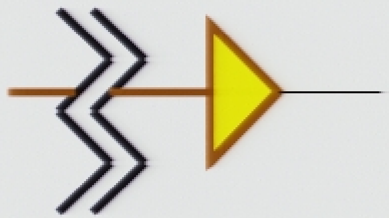


1. Insert rank 1 projectors $\sum \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.
3. Push projector back before (purified) earlier circuit.
4. Complete argument for ExRec-Cor: G_i disappears

Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.

$\Pi_i' = U \Pi_i U^{-1}$

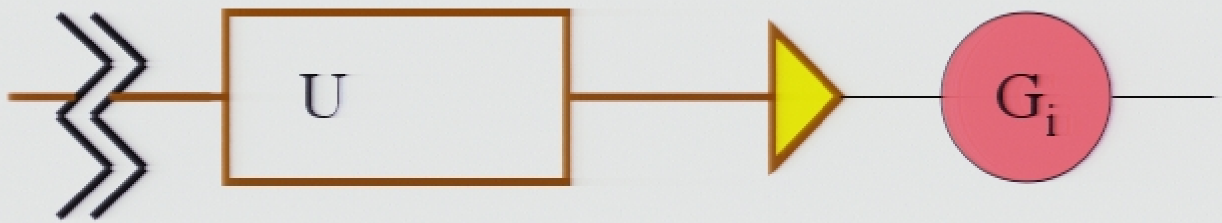


1. Insert rank 1 projectors $\sum \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.
3. Push projector back before (purified) earlier circuit.
4. Complete argument for ExRec-Cor: G_i disappears

Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.

$\Pi_i' = U \Pi_i U^{-1}$

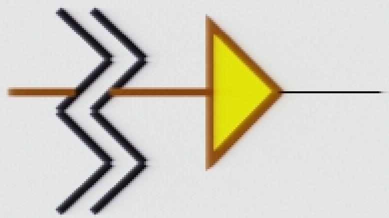


1. Insert rank 1 projectors $\sum \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.
3. Push projector back before (purified) earlier circuit.
4. Complete argument for ExRec-Cor: G_i disappears

Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.

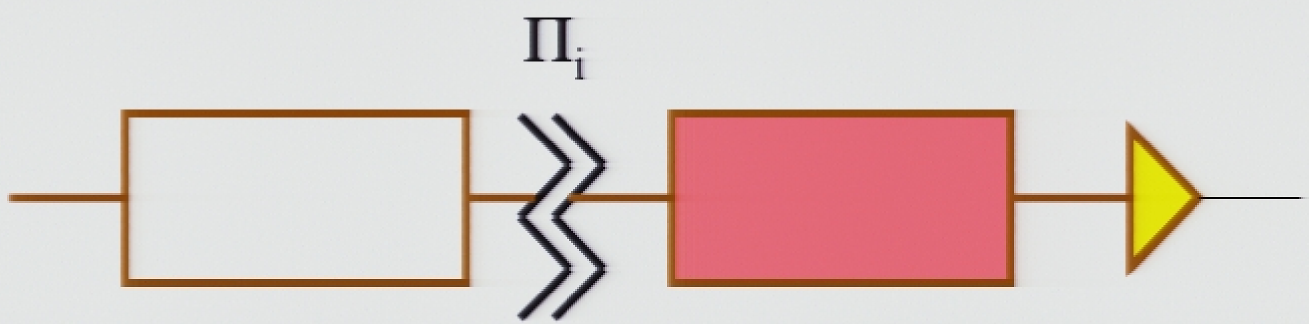
$\Pi_i' = U \Pi_i U^{-1}$



1. Insert rank 1 projectors $\Sigma\, \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.
3. Push projector back before (purified) earlier circuit.
4. Complete argument for ExRec-Cor: G_i disappears

Bad Subrectangles (Part I)

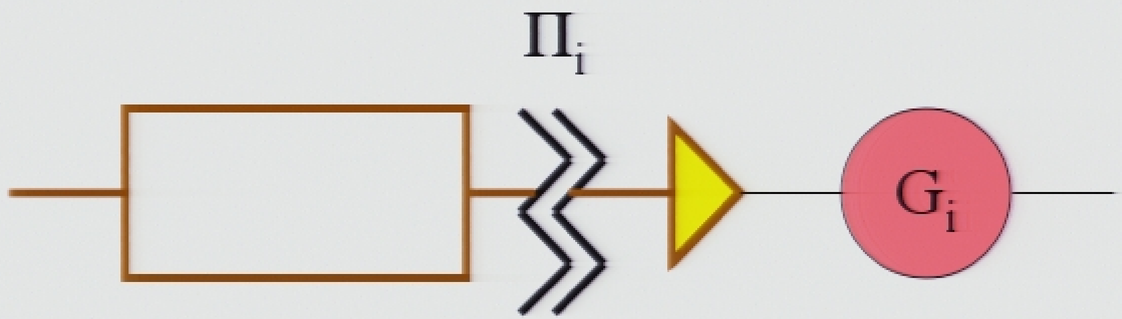
We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.



1. Insert rank 1 projectors $\sum \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.

Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.

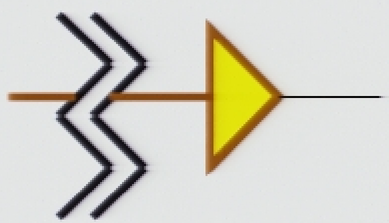


1. Insert rank 1 projectors $\Sigma \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.

Bad Subrectangles (Part I)

We don't know how to pull an ideal decoder through a bad subrectangle: there might not be a consistent rule for all input states.

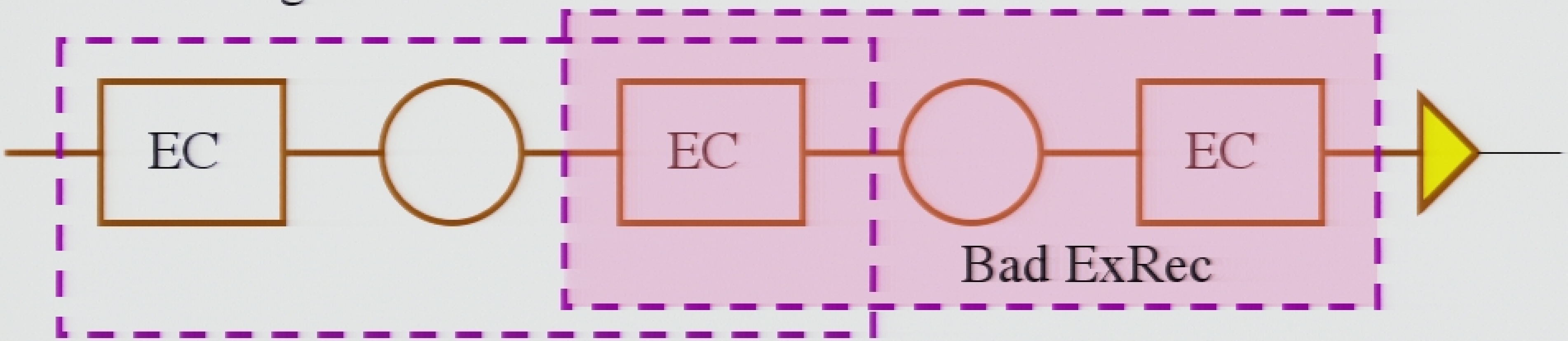
$\Pi_i' = U\Pi_i U^{-1}$



1. Insert rank 1 projectors $\sum \Pi_i = I$.
2. Push decoder back to get erroneous gate, depending on i.
3. Push projector back before (purified) earlier circuit.
4. Complete argument for ExRec-Cor: G_i disappears

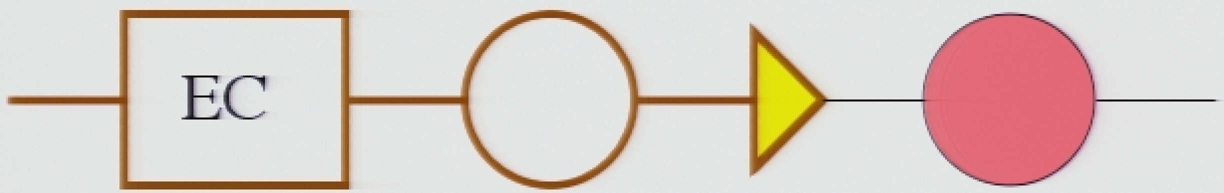
Bad Subrectangles (Part II)

Since ExRecs overlap, two consecutive ExRecs have correlated errors. However, this does not seriously affect our argument:



Bad Subrectangles (Part II)

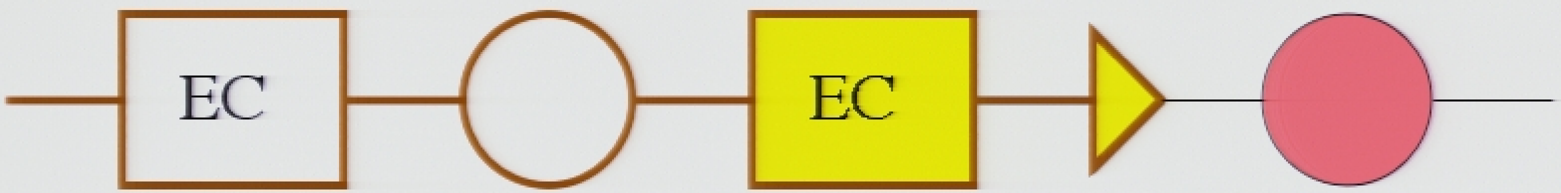
Since ExRecs overlap, two consecutive ExRecs have correlated errors. However, this does not seriously affect our argument:



1. Push decoder through later bad ExRec.

Bad Subrectangles (Part II)

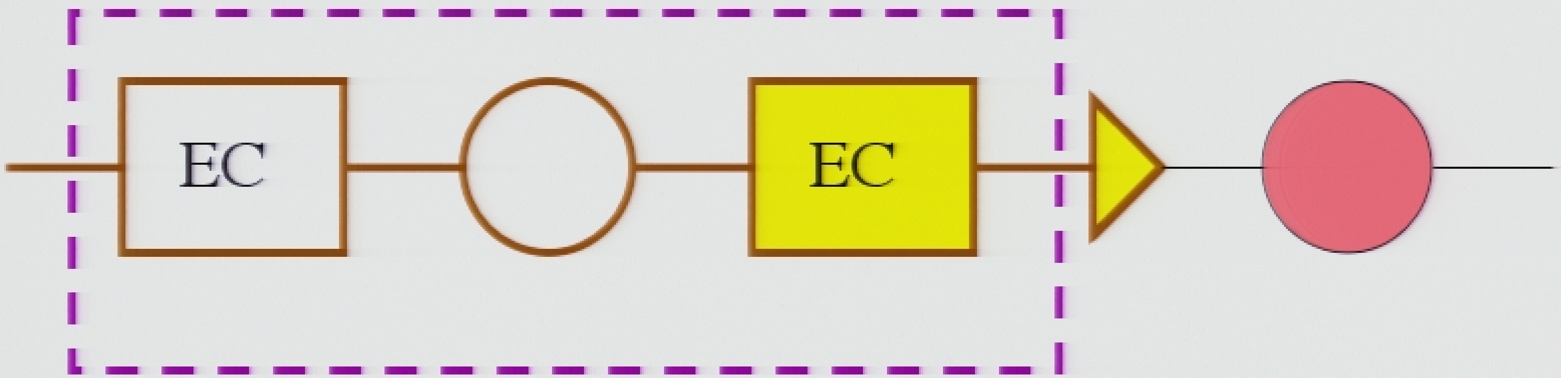
Since ExRecs overlap, two consecutive ExRecs have correlated errors. However, this does not seriously affect our argument:



1. Push decoder through later bad ExRec.
2. Insert a perfect EC before the decoder.

Bad Subrectangles (Part II)

Since ExRecs overlap, two consecutive ExRecs have correlated errors. However, this does not seriously affect our argument:



1. Push decoder through later bad ExRec.

2. Insert a perfect EC before the decoder.

3. We now have an ExRec which does not overlap with the bad ExRec - it has uncorrelated errors.

Refined Notions of Badness

We work our way backwards through a circuit. When we find a bad ExRec, remove its leading ECs from the preceeding ExRecs. An ExRec is bad in context if it has more than one error remaining after the above procedure.

Also, note that the argument for ExRec-Cor at level k only depends on ExRec-Cor being true at level 1. Therefore define a malignant set of locations as a set which causes ExRec-Cor to fail at level 1. A level 1 ExRec is bad (in context) if it has faults at a malignant set of locations, and a level k ExRec is bad (in context) if it has sub-ExRecs which are bad in context at a malignant set of locations.

Counting malignant sets of errors allows an improved threshold.

The Threshold Theorem

Theorem: There exists a threshold p_t such that, if the error rate per gate and time step is $p < p_t$, arbitrarily long quantum computations are possible with arbitrary accuracy.

Proof: At level 1, the probability of having a malignant set of locations is f(p) for some function p.

We know $f(p) = O(p^2)$, so there exists p_t such that if $p < p_t$, then $f(p) < p_t (p/p_t)^2$.

If the probability of a level k ExRec being bad is p_k, then the probability of a level k+1 ExRec being bad is at most $f(p_k)$. Thus, we find

$$p_k < p_t (p/p_t)^{2^k}$$

The computation succeeds when all top-level ExRecs are good.

What's New Here?

- Substantially simpler proof.
- Applies to codes correcting one error, and takes full advantage of codes correcting t errors.
- Counting malignant sets of locations provides framework for proving a good value of threshold.